The Astrophysical Journal Supplement Series, 190:267–274, 2010 October doi:10.1088/0067-0049/190/2/267

ALGORITHM FOR THE EVALUATION OF REDUCED WIGNER MATRICES

G. Prézeau[1,2] and M. Reinecke[3]
[1] Jet Propulsion Laboratory, 4800 Oak Grove Drive, Pasadena, CA 91109, USA
[2] California Institute of Technology, 1200 East California Boulevard, Pasadena, CA 91125, USA
[3] Max-Planck-Institut für Astrophysik, Karl-Schwarzschild-Str. 1, 85741 Garching, Germany
Received 2010 January 14; accepted 2010 August 4; published 2010 September 16

ABSTRACT

Algorithms for the fast and exact computation of Wigner matrices are described and their application to a fast and massively parallel 4π convolution code between a beam and a sky is also presented.

Key words: cosmic background radiation – cosmology: observations – instrumentation: adaptive optics – methods: data analysis – techniques: image processing

Online-only material: color figures

1. INTRODUCTION

Wigner matrices have a ubiquitous presence in science; from the computation of molecular quantum states, through the description of solitons in particle physics and the convolution of beam and sky algorithms in astronomy, they are needed to sometimes very high quantum numbers. Algorithms that calculate Wigner matrices quickly and accurately are therefore far-reaching in their applicability. Other methods have been developed that calculate these matrices exactly but with suboptimal performance to very high angular momenta (Risbo 1996), or approximately but very efficiently (Rowe et al. 2001), but none of the methods calculate them exactly and quickly to almost arbitrarily high angular momentum. Two such methods are presented in this paper and applied to a convolution algorithm between beam and sky. The following section gives some basic properties of Wigner matrices, and this is followed by a section describing the algorithm. The fourth section describes its application to convolution and a summary is presented at the end.

2. WIGNER MATRICES

Wigner matrix elements,[4] typically denoted by $D^l_{mm'}(\alpha, \beta, \gamma)$, are the eigenfunctions of the Schrödinger equation for a symmetric top and form an irreducible basis of the Lie group SU(2) and the rotation group SO(3); the angles α, β, and γ are the Euler angles that define the orientation of the top. As basis functions of SU(2), the $D^l_{mm'}(\alpha, \beta, \gamma)$ satisfy the standard angular momentum relations

$$\hat{\mathrm{J}}^2 D^l_{mm'}(\alpha, \beta, \gamma) = l(l+1) D^l_{mm'}(\alpha, \beta, \gamma), \tag{1}$$

$$\hat{\mathrm{J}}_z D^l_{mm'}(\alpha, \beta, \gamma) = m D^l_{mm'}(\alpha, \beta, \gamma), \tag{2}$$

$$\hat{\mathrm{J}}_{z'} D^l_{mm'}(\alpha, \beta, \gamma) = m' D^l_{mm'}(\alpha, \beta, \gamma), \tag{3}$$

where l labels the irreducible representation of SU(2) and also corresponds to the quantum number representing the total angular momentum of the eigenfunction; $-l \leqslant m, m' \leqslant l$ are the quantum numbers representing the projections of the total angular momentum on two z-axes rotated with respect to each other as described below.

[4] For a nice review of Wigner matrices, see Varshalovich et al. (1988).

The Euler angles are defined as three rotations: a rotation γ about the z-axis that rotates the x- and y-axes $\rightarrow x'$ and y'; this first rotation is followed by a rotation β about the new y'-axis rotating the x'- and z-axes $\rightarrow x''$ and z'; the final rotation α is about z'. In the basis we are using as defined by Equations (2) and (3), the operators $\hat{\mathrm{J}}_z$ and $\hat{\mathrm{J}}_{z'}$ are diagonal and $D^l_{mm'}(\alpha, \beta, \gamma)$ has the form

$$D^l_{mm'}(\alpha, \beta, \gamma) = e^{-im\alpha} d^l_{mm'}(\beta) e^{-im'\gamma}, \tag{4}$$

where

$$d^l_{mm'}(\beta) = \langle lm| \exp\left[-i\frac{\beta}{\hbar}\hat{\mathrm{J}}_y\right] |lm'\rangle. \tag{5}$$

Here, $d^l_{mm'}(\beta)$ is called the reduced Wigner matrix element and consists of the overlap of a spherical harmonic with another spherical harmonic that has been rotated by an angle β about the y-axis. The differential equation satisfied by $d^l_{mm'}(\beta)$ is

$$\frac{d^2 d^l_{mm'}(\beta)}{d\beta^2} + \cot\beta \frac{d d^l_{mm'}(\beta)}{d\beta} + \left(\frac{2mm'\cos\beta - m^2 - m'^2}{\sin^2\beta} + l(l+1)\right) d^l_{mm'}(\beta) = 0. \tag{6}$$

From the Schrödinger equation in Equation (6), it is possible to extract three-term recursion relations that relate reduced Wigner matrix elements that differ in their quantum numbers. In principle, it is possible to use such relations to calculate the $d^l_{mm'}(\beta)$. Three-term recursion relations can be unstable, which limits their usefulness unless the potential pitfalls are identified and avoided. Two examples of these relations are

$$\frac{-m + m'\cos\beta}{\sin\beta} d^l_{mm'}(\beta) = \frac{1}{2}\sqrt{(l+m')(l-m'+1)} d^l_{mm'-1}(\beta) + \frac{1}{2}\sqrt{(l-m')(l+m'+1)} d^l_{mm'+1}(\beta) \tag{7}$$

and

$$\left[\cos\beta - \frac{mm'}{l(l+1)}\right] d^l_{mm'}(\beta) = \frac{\sqrt{(l^2-m^2)(l^2-m'^2)}}{l(2l+1)} d^{l-1}_{mm'}(\beta) + \frac{\sqrt{[(l+1)^2-m^2][(l+1)^2-m'^2]}}{(l+1)(2l+1)} d^{l+1}_{mm'}(\beta). \tag{8}$$

Generally, three-term recursion relations will have two linearly independent solutions, f_n and g_n (Press et al. 1988); these solutions can be oscillatory or exponentially decreasing or increasing. In the non-oscillatory case, f_n is the *minimal* solution if

$$\frac{f_n}{g_n} \to 0 \text{ as } n \to \infty, \tag{9}$$

while g_n is the *dominant* solution. For solutions to the Schrödinger equation, exponentially increasing/decreasing solutions appear only in the region where a particle cannot classically exist because of energy conservation, but where a wave function can be non-zero in quantum mechanics. In the case of a rigid rotor (Edmonds 1957), the kinetic energy of a spherically symmetric rotor is

$$2IT = p_\beta^2 + \frac{1}{\sin^2\beta}\left(p_\gamma^2 + p_\alpha^2 - 2p_\alpha p_\gamma \cos\beta\right). \tag{10}$$

In classical mechanics, $p_\beta^2 > 0$. In quantum mechanics, the quantization of Equation (10) means substituting $p_\alpha \to -i\partial/\partial\alpha$, $p_\beta \to -i\partial/\partial\beta$, and $p_\gamma \to -i\partial/\partial\gamma$. These substitutions combined with an eigenfunction of the form (4) and the additional substitution $2IT \to l(l+1)D^l_{mm'}(\alpha,\beta,\gamma)$ inferred from Equation (1) yields Equation (6). Since p_β^2 corresponds to the first two terms of Equation (6), one concludes that classically we would have

$$l(l+1) + \frac{2mm'\cos\beta - m^2 - m'^2}{\sin^2\beta} \geqslant 0. \tag{11}$$

Wherever Equation (11) is not satisfied, the solutions will be exponentially suppressed or divergent. When solving the Schrödinger equation for the physical solutions, the divergent solutions are simply put to zero. When using the three-term recursion relations, the divergent solution can be "sniffed" out because of round-off errors and the recursions quickly fail. One special case where this cannot happen is when $m, m' = 0$ where Equation (11) is always satisfied since $l \geqslant 0$. In that case, Equation (8) is stable and can be used to calculate $d^l_{00}(\beta)$ to very high l extremely accurately.

For the cases where $m, m' \neq 0$, we can still use three-term recursion relations provided we do so in the right direction in the quantum number being varied. For example, looking at Equation (7), one can either calculate each $d^l_{mm'}(\beta)$ for increasing m' or decreasing m'. In one direction, the divergent solution will be growing while in the other it will be shrinking. To determine the direction in which Equation (7) is stable, one only needs to consider Equation (11). Assume you are interested in evaluating all the reduced Wigner matrix elements $d^l_{0m'}(\beta)$ for $0 \leqslant m' \leqslant l$ using Equation (7); you can choose to begin your recurrence with $d^l_{00}(\beta)$ and increasing m' or begin from $d^l_{0l}(\beta)$ and decreasing m'. To use Equation (7) in a stable manner, you need to start from $d^l_{0l}(\beta)$ and decrease m'. Putting $m = 0$ in Equation (11) yields the new condition

$$l(l+1) - \frac{m'^2}{\sin^2\beta} \geqslant 0, \tag{12}$$

where it is seen that as m' increases from 0 (taking, for example, $\beta = \pi/4$), we approach the non-classical region and violate Equation (12) at $m' \geqslant \sin\beta\sqrt{l(l+1)}$; increasing m' further means sampling the divergent dominant solution of the Schrödinger equation from which Equation (7) is derived. It is then clear that the stable direction to use Equation (7) is for decreasing $|m'|$. From Equation (11), it is seen quite generally that the recursion relations (7) and (8) will be stable provided they are used in the direction of decreasing $|m'|$ and increasing l, respectively.

In addition to Equations (7) and (8), a third recursion relation in β can be derived by discretizing the derivatives in Equation (6) with the relations

$$f'(x) \cong \frac{f(x+\epsilon) - f(x-\epsilon)}{2\epsilon} + O(\epsilon^2 f'''), \tag{13}$$

$$f''(x) \cong \frac{f(x+\epsilon) + f(x-\epsilon) - 2f(x)}{\epsilon^2} + O(\epsilon f'''). \tag{14}$$

Substituting into Equation (6) yields

$$\left[\epsilon^2\left(\frac{2mm'\cos\beta - m^2 - m'^2}{\sin^2\beta} + l(l+1)\right) - 2\right] d^l_{mm'}(\beta)$$
$$\cong \left(\frac{\epsilon\cot\beta}{2} - 1\right) d^l_{mm'}(\beta-\epsilon) - \left(\frac{\epsilon\cot\beta}{2} + 1\right) d^l_{mm'}(\beta+\epsilon)$$
$$+ O(\epsilon^3 d^{l'''}_{mm'}(\beta)). \tag{15}$$

From Equation (11), it is seen that this recursion relation should be used for increasing β if $0 < \beta < \pi/2$ and decreasing β if $\pi/2 < \beta < \pi$. Examples of these conclusions are given in Figure 1. The top plot shows the change in the behavior of $d^l_{mm'}(\beta)$ with increasing l as one moves from the non-classical to classical regions; in that case, the angle β was chosen so that Equation (11) is satisfied only when $l \geqslant 100$. The middle plot shows the variation of $d^l_{mm'}(\beta)$ with m'. With $\beta = 0.52302$, $l = 500$, and $m = 0$, the transition from non-classical to classical regimes occurs at $m' = 250$. The last plot shows the variation of $d^l_{mm'}(\beta)$ with β; the value of $m' = 71$ was chosen so that the transition from non-classical to classical occurs at $\beta = \pi/4$. A noticeable feature of all three plots is that the tallest peak is always the first peak after the transition to the classical region. This is qualitatively understandable from Equation (5) where the reduced Wigner matrix is seen to characterize the overlap between two spin states after a rotation. In the classical limit of large l, the angle ω of the spin direction of a quantum object with the z-axis is given by

$$|lm\rangle\colon \cos\omega \approx \frac{m}{\sqrt{l(l+1)}}. \tag{16}$$

One might expect that the overlap would be greatest when the "classical" spins are aligned after the rotation about the y-axis. From Equations (11) and (16), we can show that this is the case when

$$\beta = \operatorname{acos}\left(\frac{m'}{\sqrt{l(l+1)}}\right) - \operatorname{acos}\left(\frac{m}{\sqrt{l(l+1)}}\right). \tag{17}$$

In our example, $m = 0$ and Equation (17) reads $\sin^2(\beta) = m'^2/[l(l+1)]$ and the overlap is greatest at the transition point.

3. ALGORITHM

The evaluation of the $d^l_{mm'}(\beta)$ from Equations (7) and (8) requires starting values for the recursions. For large l, however, those values are often vanishingly small and cannot be represented by any of the IEEE 754 floating-point data formats which are used on practically all current computer hardware. Starting the recursions with 0 and 1 does not help because there will come a point where the $d^l_{mm'}(\beta)$ become too big to be represented numerically. Two solutions to this problem are presented here.

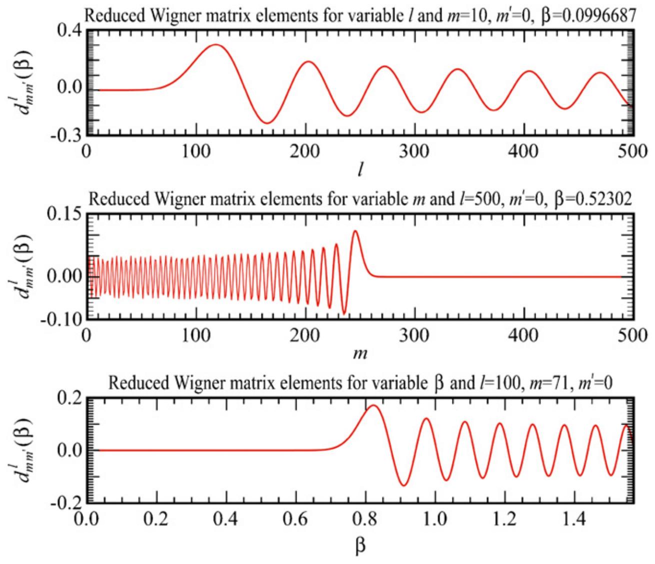

Figure 1. Top plot shows the variation of $d^l_{010}(0.0996687)$ for $10 \leqslant l \leqslant 500$, the middle plot shows the variation of $d^{500}_{0m'}(0.52331)$ for $0 \leqslant m' \leqslant 500$, and the bottom plot shows the variation of $d^{100}_{071}(\beta)$ for $0 \leqslant \beta \leqslant \pi/2$.
(A color version of this figure is available in the online journal.)

3.1. $d^l_{mm'}(\beta)$ Ratios

From Equation (6) and the plots of Figure 1, it is clear that the $d^l_{mm'}(\beta)$ vary smoothly with varying l, m, and β. As a result, a recursion relation of ratios should always be finite in the non-classical region where the $d^l_{mm'}(\beta)$ are not oscillatory, and one only has to worry about singularities in the ratios in the classical/oscillatory region where the denominators could vanish if evaluated at a zero of the $d^l_{mm'}(\beta)$. In this ratio-based method, Equations (7) and (8) can be rewritten as

$$\frac{d^l_{mm'}}{d^l_{mm'-1}} = \frac{\sqrt{(l+m')(l-m'+1)}}{m\,\mathrm{cosec}\beta - m'\mathrm{cotan}\beta - \sqrt{(l-m')(l+m'+1)}\frac{d^l_{mm'+1}}{d^l_{mm'}}}, \tag{18}$$

$$\frac{d^l_{mm'}}{d^{l+1}_{mm'}} = \frac{(l+1)\sqrt{(l^2-m'^2)(l^2-m^2)}}{(2l+1)(mm'-\cos\beta) - l\sqrt{[(l+1)^2-m'^2][(l+1)^2-m^2]}\frac{d^{l-1}_{mm'}}{d^l_{mm'}}}. \tag{19}$$

Using

$$\frac{d^l_{ml}}{d^l_{ml-1}} = \frac{\sqrt{l/2}\sin\beta}{l\cos\beta - m} \text{ and} \tag{20}$$

$$\frac{d^l_{ml}}{d^{l+1}_{ml}} = \sqrt{\frac{(l+m+1)(l-m+1)}{2l+1}}[(l+1)\cos\beta - m]^{-1}, \tag{21}$$

the ratios can be calculated recursively down to $m' = 0$ if using Equation (18) or up to $l = l_{\max}$ if using Equation (19). For example, in the case where all the $d^l_{0m'}$ for $l \geqslant m' \geqslant 0$ are required, one would start with Equation (18) to calculate

$$\frac{d^l_{0l}(\beta)}{d^l_{0l-1}(\beta)}, \frac{d^l_{0l-1}(\beta)}{d^l_{0l-2}(\beta)}, \ldots, \frac{d^l_{02}(\beta)}{d^l_{01}(\beta)}, \frac{d^l_{01}(\beta)}{d^l_{00}(\beta)}. \tag{22}$$

Then, to calculate the $d^l_{0m'}$, one would need to know d^l_{00}. Fortunately, the d^l_{00} are easy to calculate because their recursion relation does not contain exponential solutions as remarked under Equation (11):

$$\cos\beta d^l_{00}(\beta) = \frac{l}{2l+1}d^{l-1}_{00} + \frac{l+1}{2l+1}d^{l+1}_{00}. \tag{23}$$

Once $d^l_{00}(\beta)$ has been calculated, d^l_{01} can be calculated from the ratios; in order, each $d^l_{0m}(\beta)$ can be calculated by multiplying adjacent ratios until $d^l_{0l}(\beta)$ has been evaluated. In the case where $d^l_{1m'}$ for $l \geqslant m' \geqslant 0$ are also needed, the set of ratios

$$\frac{d^l_{1l}(\beta)}{d^l_{1l-1}(\beta)}, \frac{d^l_{1l-1}(\beta)}{d^l_{1l-2}(\beta)}, \ldots, \frac{d^l_{12}(\beta)}{d^l_{11}(\beta)}, \frac{d^l_{11}(\beta)}{d^l_{10}(\beta)} \tag{24}$$

is computed next. To then calculate the $d^l_{1m'}$, one needs to know d^l_{10}. Fortunately, $d^l_{01} = -d^l_{10}$ has previously been calculated and all the $d^l_{1m'}$ can be obtained up to d^l_{1l}. In this fashion, all the $d^l_{mm'}(\beta)$ can be calculated up to a desired $m = m_{\max}$.

The special and extremely rare case where a ratio $d^l_{mm'+1}/d^l_{mm'}$ is infinite can be handled by using Equation (7) where $d^l_{mm'}(\beta)$ is set to zero and substituting an infinite ratio and a null ratio for a single finite ratio:

$$\frac{d^l_{mm'+1}}{d^l_{mm'}}, \frac{d^l_{mm'}}{d^l_{mm'-1}} \rightarrow \frac{d^l_{mm'+1}}{d^l_{mm'-1}} = -\frac{\sqrt{(l+m')(l-m'+1)}}{\sqrt{(l-m')(l+m'-1)}}. \tag{25}$$

Note that in contrast to the method described in Risbo (1996), the column of matrix elements $d^l_{ml}(\beta) \rightarrow d^l_{m0}(\beta)$ can be evaluated without having to calculate every single $d^{l'}_{mm'}(\beta)$ for $l' < l$. The same tricks can be applied to the recursion relation in l for the evaluation of $d^l_{lm'}(\beta) \rightarrow d^{l_{\max}}_{lm'}(\beta)$. First calculate the column of elements $d^{l_{\max}}_{l l_{\max}}(\beta) \rightarrow d^{l_{\max}}_{l0}(\beta)$ and then calculate the ratios

$$\frac{d^l_{lm'}(\beta)}{d^{l+1}_{lm'}(\beta)}, \frac{d^{l+1}_{lm'}(\beta)}{d^{l+2}_{lm'}(\beta)}, \ldots, \frac{d^{l_{\max}-2}_{lm'}(\beta)}{d^{l_{\max}-1}_{lm'}(\beta)}, \frac{d^{l_{\max}-1}_{lm'}(\beta)}{d^{l_{\max}}_{lm'}(\beta)}. \tag{26}$$

Knowing $d^{l_{\max}}_{lm'}(\beta)$ allows us to evaluate $d^{l_{\max}-1}_{lm'}(\beta) \rightarrow d^l_{lm'}(\beta)$.

For a particular β, it is sufficient to compute the elements of $d^l_{mm'}(\beta)$ for $0 \leqslant m \leqslant l$ and $-l \leqslant m' \leqslant l$ to know the entire matrix $d^l(\beta)$. To fill out the rest of the matrix, the symmetry relations in the Appendix can be used.

3.2. Wigner Matrix Elements by l-Recursion

Another way to deal with the underflow problem is to start from Equation (8) with the following initialization values:

$$d^l_{l,m}(\beta) = A\left(\cos\frac{\beta}{2}\right)^{l+m}\left(-\sin\frac{\beta}{2}\right)^{l-m}, \tag{27}$$

$$d^l_{-l,m}(\beta) = A\left(\cos\frac{\beta}{2}\right)^{l-m}\left(\sin\frac{\beta}{2}\right)^{l+m}, \tag{28}$$

$$d^l_{m,l}(\beta) = A\left(\cos\frac{\beta}{2}\right)^{l+m}\left(\sin\frac{\beta}{2}\right)^{l-m}, \tag{29}$$

$$d^l_{m,-l}(\beta) = A \left(\cos\frac{\beta}{2}\right)^{l-m} \left(-\sin\frac{\beta}{2}\right)^{l+m}, \tag{30}$$

where $A = \sqrt{(2l)!/(l+m)!(l-m)!}$. As far as Equations (27)–(30) are concerned, the underflow problem can be avoided by simply calculating the logarithm of the absolute value of the matrix element and storing its sign separately. Equation (27), e.g., then transforms to

$$\begin{aligned}\ln\left|d^l_{l,m}(\beta)\right| = 0.5\,(\ln((2l)!) - \ln((l+m)!) - \ln((l-m)!)\\ + (l+m)\ln|\cos(\beta/2)| + (l-m)\ln|\sin(\beta/2)|\,.\end{aligned} \tag{31}$$

In cases where one of the last two terms is $-\infty$, the recursion in l can be stopped immediately, since all subsequent values will be zero.

The logarithms of the faculties are easily precomputed, so that the seed value for the recursion can be obtained in $\mathcal{O}(1)$ operations.

Since the result of Equation (31) is in some circumstances much smaller than the individual terms on the right-hand side, cancellation errors may reduce the number of significant digits of the result. In order to have the highest accuracy that can be achieved without sacrificing too much performance, the computation of the seed value is carried out with extended IEEE precision (corresponding to the C++ data type `long double`).

The recursion relation (8) itself unfortunately cannot be computed conveniently in logarithms; therefore, a way must be found to represent floating-point values with an extreme dynamic range, which does not incur a high performance penalty.

This was implemented by representing a floating-point number v using an IEEE double-precision value d and an integer scale n, such that

$$v = d \cdot S^n, \tag{32}$$

where either $d = 0$, or $S^{-1} \leqslant |d| \leqslant S$ and S (the "scale factor") is a positive constant that can be represented as a double-precision IEEE value. Using this prescription, v does not have a unique representation as a (d, n)-pair, but this is not a problem.

Similar techniques have been in use since at least three decades in numerical algorithms; for a recent example, see the spherical harmonic transform routines of the HEALPix package.

It is advantageous to choose a scale factor which is an integer power of 2, because multiplying or dividing by such a factor only affects the exponent of a floating-point value stored in binary format, and is therefore exact (ignoring possible under- or overflows). In order to avoid frequent re-scaling of d, the scale factor should also be rather large; the value adopted for our implementation is 2^{90}.

Using this representation for the $d^l_{mm'}(\beta)$, the recursion is performed until either $l_{\max}$ is reached, or the matrix element has become large enough to be safely represented by a normal double-precision variable (the threshold value used in the code is 2^{-900}). In the latter case, the remaining computations up to $l_{\max}$ are done with standard floating-point arithmetic, which is significantly faster.

4. CONVOLUTION

One area where fast and efficient techniques of computing $d^l_{mm'}(\beta)$ are particularly valuable is in 4π convolution (Wandelt & Gorski 2001). For the convolution of two fields $b(\Omega)$ and $s(\Omega)$ defined on a sphere, the following integral must be calculated:

$$c = \int d\Omega b^*(\Omega)s(\Omega). \tag{33}$$

In the physical application where $b(\Omega)$ is a beam from a horn located on a slowly rotating space telescope that scans the sky (denoted $s(\Omega)$) as it orbits the Sun (e.g., *WMAP* or *Planck* missions), a large number of convolutions must be performed to account for every possible orientation (α, β, γ) of the satellite

$$c(\Omega') = \int d\Omega[\mathrm{R}(\Omega')b(\Omega)]^* s(\Omega), \tag{34}$$

where $\mathrm{R}(\Omega')$ is a rotation matrix that rotates the beam to a particular orientation of the satellite and is defined

$$\mathrm{R}(\alpha,\beta,\gamma)\mathrm{Y}_{lm}(\theta,\phi) = \sum_{m'=-l}^{l} D^l_{m'm}(\alpha,\beta,\gamma)\mathrm{Y}_{lm'}(\theta,\phi), \tag{35}$$

and where the $\mathrm{Y}_{lm}(\theta,\phi)$ are spherical harmonics. The $\mathrm{Y}_{lm}(\theta,\phi)$ are related to the $D^l_{mm'}(\alpha,\beta,\gamma)$ through the relation

$$\mathrm{Y}_{lm}(\theta,\phi) = (-1)^m\sqrt{\frac{2l+1}{4\pi}}\,D^l_{0m}(0,\theta,\phi) \tag{36}$$

and can therefore be calculated using the methods described above.

For beams with significant side-lobes stemming from reflections of light far away from the line of sight as is the case for both the *WMAP* and *Planck* missions, the beams can cover a significant portion of the sky and full-sky convolutions are necessary; as shown in Wandelt & Gorski (2001), such full-sky convolutions are much faster when performed in harmonic space instead of pixel space. We now describe a very fast and massively parallel method to perform full-sky convolutions in harmonic space.

4.1. `Conviqt`

`Conviqt` (CONvolution VIa the Quantum Top equation) is a fast 4π convolution algorithm that relies on fast computational methods for reduced Wigner matrix elements. Starting from Equations (34) and (35), the beam and sky fields can be expanded on the spherical harmonic basis to yield

$$c(\alpha,\beta,\gamma) = \sum_{m_{\mathrm{b}}=-m_{b\max}}^{m_{b\max}} \sum_{m_{\mathrm{sky}}=-l_{\max}}^{l_{\max}} e^{im_{\mathrm{b}}\alpha} e^{im_{\mathrm{sky}}\gamma} C_{m_{\mathrm{b}}m_{\mathrm{sky}}}(\beta), \tag{37}$$

$$C_{m_{\mathrm{b}}m_{\mathrm{sky}}}(\beta) \equiv \sum_{l=0}^{l_{\max}} b^*_{lm_{\mathrm{b}}} d^l_{m_{\mathrm{b}}m_{\mathrm{sky}}}(\beta) s_{lm_{\mathrm{sky}}}, \tag{38}$$

where $b_{lm_{\mathrm{b}}}$ and $s_{lm_{\mathrm{sky}}}$ are the spherical components of the beam and sky fields.[5] Typically, the $b_{lm_{\mathrm{b}}}$ are negligible for some $m_{b\max} < m_{\mathrm{b}} \ll l_{\max}$. Noting that the number of β angles needed for the convolution scales as $l_{\max}$, the evaluation of Equation (37) scales as $O(l^2_{\max} m_{b\max} \log(l_{\max}))$ after the use

[5] For ease of reading, only the scalar case is described since the generalization to polarized maps and beams is easily accomplished by evaluating Equation (38) for the additional pairs $(b^G_{lm_{\mathrm{b}}}, s^G_{lm_{\mathrm{sky}}})$ and $(b^C_{lm_{\mathrm{b}}}, s^C_{lm_{\mathrm{sky}}})$.

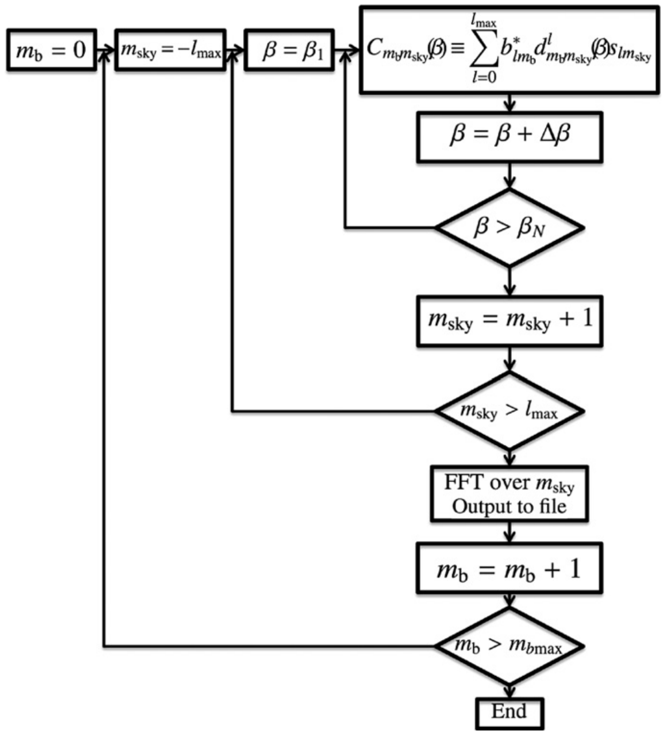

Figure 2. Calculation of the $C_{m_b m_{sky}}(\beta)$ for an interval of angles $\beta_1 \leqslant \beta \leqslant \beta_N$ requires three nested loops over β, m_{sky}, and m_b as shown above.

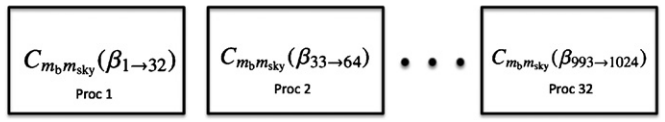
$C_{m_b m_{sky}}(\beta_{1\to 32})$
Proc 1
$C_{m_b m_{sky}}(\beta_{33\to 64})$
Proc 2
$C_{m_b m_{sky}}(\beta_{993\to 1024})$
Proc 32

Figure 3. Generic parallelization of `conviqt` where the convolution is performed on 32 processors and the range $[0, \pi/2]$ is split into $N-1$ intervals with $N = 1024$. The first 32 $C_{m_b m_{sky}}(\beta)$ are calculated on processor 1 for 32 β's with $0 \leqslant \beta \leqslant 31 \times \pi/(N-1)$, and so on for each processor. Note that the range $\pi/2 < \beta \leqslant \pi$ can be obtained from the symmetries of the $d^l_{mm'}(\beta)$.

of the Fast Fourier Transform algorithm to perform the summations. The numerically expensive part of Equation (37) is the computation of Equation (38) which scales as $O(l^3_{max} m_{bmax})$. Two separate computations of Equation (38) scale as $O(l^3_{max} m_{bmax})$: the computation of the $d^l_{m_b m_{sky}}(\beta)$, and the evaluation of the sum over $0 \leqslant l \leqslant l_{max}$ for every single m, m_{sky}, and β which form nested loops as seen in Figure 2. The fast methods described in the previous section are used to compute the $d^l_{m_b m_{sky}}(\beta)$. To evaluate the $C_{m_b m_{sky}}(\beta)$ for each β, a massively parallel MPI-based approach is used since the $C_{m_b m_{sky}}(\beta)$ are uncorrelated between the different β and can be computed by different tasks as shown in Figure 3. As seen in Section 4.3.2, `conviqt` scales very well with the number of processors and its scaling can be described as

$$\texttt{conviqt scaling} \sim l^3_{max} m_{bmax}/n, \tag{39}$$

where n is the number of processors. Additional acceleration techniques for both the computation of the $d^l_{m_b m_{sky}}(\beta)$ and for the evaluation of the sums over l are described in the following subsection.

4.2. *Acceleration Techniques for the $d^l_{m_b m_{sky}}(\beta)$*

In simulations of the measurement of cosmic microwave background, convolutions appear repeatedly especially if Monte Carlos are required. Since the generation of Wigner matrix elements is typically the most computationally intensive part of the convolution algorithm, a large effort was made to increase its efficiency. This has two aspects: first to compute the matrix elements as quickly as possible, but also to decide (if possible) which matrix elements are too small to contribute measurably to the result and skip their generation altogether.

4.2.1. *Skipping Unneeded Calculations*

When performing convolutions, especially within the context of Monte Carlo simulations where many convolutions with the same l_{max} and m_{bmax} are needed, it is computationally profitable to skip unneeded calculations.[6] Some terms in the sum of Equation (38) need not be included because their $d^l_{m_b m_{sky}}(\beta)$ are vanishingly small. To determine which terms to exclude, we turn to Equation (11) where three general possibilities are considered:

1. m_b and m_{sky} are of similar magnitude and much smaller than l.
2. m_b and m_{sky} are of similar magnitude and of the same order of magnitude as l.
3. m_b and m_{sky} are of widely differing magnitude with one much smaller than l and one of similar magnitude.

In each of these possibilities, the neglected $d^l_{m_b m_{sky}}(\beta)$ are those evaluated at β angles that correspond to the non-classical, exponentially suppressed region. Before explaining how these $d^l_{m_b m_{sky}}(\beta)$ are identified, `conviqt`'s nested structure should be described. In `conviqt`, the outermost loop deals with m_b (which ranges from 0 to m_{bmax}; nested into the m_b-loop is the m_{sky}-loop ranging from $-m_{sky}$ to m_{sky}; nested in the m_{sky}-loop is the loop over the β processed by that particular task.[7] Finally, the innermost loop is that over l which is where the condition is applied. To derive the minimum l such that outside the parameter space defined by $l_{min} \leqslant l \leqslant l_{max}$, $d^l_{m_b m_{sky}}(\beta)$ is negligible, we use Equation (11) to write

$$\begin{aligned} l_{min} &= -\frac{1}{2} + \frac{1}{2}\left[1 + \frac{4}{\sin^2\beta}\left(m^2_{sky} + m^2_b - 2m_{sky}m_b\cos\beta\right)\right]^{1/2} \\ &\cong \frac{\sqrt{m^2_{sky} + m^2_b - 2m_{sky}m_b\cos\beta - \texttt{offset}}}{\sin\beta}, \end{aligned} \tag{40}$$

where `offset`>0 and ensures that the $d^l_{m_b m_{sky}}(\beta)$ neglected are well within the non-classical region and suppressed. Calling m_{big} the larger of $|m_b|$ or $|m_{sky}|$, it is noted that $l \geqslant m_{big}$. To determine the `offset`, we go back to the three possibilities listed above; of those, l_{min} will generally equal m_{big} in the cases where m_b and m_{sky} are of similar magnitude; only in the third case will we generally have $l_{min} > m_{big}$, namely when m_b and m_{sky} are of widely differing magnitude. From Rowe et al. (2001), this case can approximately be written as a harmonic oscillator wave function:

$$d^l_{m_b m_{sky}}(\beta) \to (-1)^{l-m_{sky}}(\sqrt{l}\sin\beta_m)^{-1/2} u_{l-m_{sky}}(\sqrt{l}(\beta - \beta_m)) \tag{41}$$

$$u_\nu(x) = (\sqrt{\pi}2^\nu \nu!)^{-1/2} H_\nu(x) e^{-x^2/2}, \tag{42}$$

[6] Note that it is generally not more efficient to evaluate all of the $d^l_{m_b m_{sky}}(\beta)$ beforehand because of the disk space required and the large amount of time needed to read them in; it is more efficient to calculate them on the fly.

[7] A note to remind the reader that `conviqt` is parallelized in β; each task will perform the convolutions in a subset of all the complete set of β where $C_{m_b m_{sky}}(\beta)$ must be calculated.

where $H_\nu(x)$ is a Hermite polynomial and $\cos\beta_m = m_b/l$. Since we are dealing with orders of magnitude, it is not necessary to exactly evaluate Equation (41) to determine `offset`, only to calculate an estimate from the factor $\exp(-x^2/2)$. We have found that using `offset`$\geqslant l_{\max}/20$ gives extremely accurate results. Finally, it is noted that $d^l_{m_b m_{\rm sky}}(0) = \delta_{m_b m_{\rm sky}}$ and for that special case we put $l_{\min} > l_{\max}$ when $m_b \neq m_{\rm sky}$ and no sum over l is performed. Thus, Equation (40) is used to estimate whether the absolute values of all $d^l_{m_b m_{\rm sky}}(\beta)$ for a given combination of l, m_b, $m_{\rm sky}$, and β lie below a certain threshold; if this is the case, the generation of these values can be skipped entirely. In particular, the most efficient version of the code written was one where the $d^{l_{\min}}_{m_b m_{\rm sky}}$ and $d^{l_{\min}+1}_{m_b m_{\rm sky}}$ were precalculated for $0 \leqslant m_b \leqslant m_{b\max}$, $-l_{\max} \leqslant m_{\rm sky} \leqslant l_{\max}$, and β subset for a particular task, and read-in as seeds to the recursion relation of Equation (8). That way, none of the unneeded $d^l_{m_b m_{\rm sky}}(\beta)$ were calculated during the convolution. This required an extra code to precompute the $d^l_{m_b m_{\rm sky}}(\beta)$. In the end, we opted for a single code based on the evaluation of the reduced Wigner matrix elements as described in Section 3.2 because of the low overhead and maintenance as well as the high efficiency.

In this approach, we calculate all the $d^l_{m_b m_{\rm sky}}(\beta)$ on the fly, but only include the relevant $d^l_{m_b m_{\rm sky}}(\beta)$ in the final l-loop. This code is self-contained and easier to maintain at a very minimal cost in performance. As the $d^l_{m_b m_{\rm sky}}(\beta)$ recursion is performed, the code checks the absolute values of the generated $d^l_{mm'}(\beta)$ and records the l index at which a predefined threshold ε (typically set to 10^{-30}) is crossed for the first time. Due to the limited dynamic range of IEEE data types, values below this threshold have no measurable influence on the convolution result and can therefore be neglected during the final summation loop, which saves a significant amount of CPU time.

4.2.2. *Precomputed Values*

In this single-code approach, a precomputation strategy well suited to the loop structure was adopted.

1. At the beginning of each run, we compute just once $\ln(\cos(\beta/2))$, $\ln(\sin(\beta/2))$, and $\cos\beta$ for all β at which we need the Wigner matrix, and as mentioned above, $\ln n!$ up to $n = 2l_{\max}$.
2. Also at the beginning we compute the tables
$$P_i = \sqrt{1/(i+1)} \quad \text{and} \quad Q_i = \sqrt{i/(i+1)}$$
for i in the range of 0 to $2l_{\max}+1$, which are needed for the next precomputation step.
3. Inside the second loop (i.e., for every combination of m_b and $m_{\rm sky}$), we compute the tables
$$\begin{aligned} F_{0,l} &= (l+1)(2l+1)P_{l+m_b}P_{l-m_b}P_{l+m_{\rm sky}}P_{l-m_{\rm sky}}, \\ F_{1,l} &= m_b m_{\rm sky}/(l(l+1)), \text{ and} \\ F_{2,l} &= Q_{l+m_b}Q_{l-m_b}Q_{l+m_{\rm sky}}Q_{l-m_{\rm sky}}(l+1)/l. \end{aligned}$$

After all these preparations, Equation (8) boils down to
$$d^{l+1}_{m_b m_{\rm sky}}(\beta) = F_{0,l}(\cos\beta - F_{1,l})d^l_{m_b m_{\rm sky}}(\beta) - F_{2,l}d^{l-1}_{mm_{\rm sky}}(\beta), \quad (43)$$
which corresponds to only five quick-to-compute floating-point operations.

The space overhead for the additional tables is $\mathcal{O}(l_{\max})$, which is insignificant compared to the $\mathcal{O}(l^2_{\max})$ memory requirement of the whole convolution code. This also means that for the reasonable assumption of $l_{\max} \lesssim 10^4$ all data required for the recursion fit conveniently into current processors' Level-2 caches.

4.2.3. *Use of $d^l_{mm'}(\beta)$ Symmetries*

The use of the $d^l_{mm'}(\beta)$ symmetries considerably cut the computational cost of the full sky convolution. In particular, Equation (A6) relates the computed values of $C_{m_b m_{\rm sky}}(\beta)$ at $\beta < \pi/2$ to those at $\beta > \pi/2$. In Equation (A6), the phase factor $(-1)^l$ is accounted for by splitting the sum in Equation (38) into even and odd l. The phase factor $(-1)^{-m'}$ is accounted for by using the relations
$$b_{lm} = (-1)^m b^*_{l-m},\ s_{lm} = (-1)^m s^*_{l-m} \quad (44)$$
and further splitting the odd and even sums of Equation (38) into real and imaginary parts. In addition, the symmetry of Equations (A1) and (44) can be used to show
$$C_{-m_b - m_{\rm sky}}(\beta) = C_{m_b m_{\rm sky}}(\beta)^* \quad (45)$$
speeding up the computation of $C_{m_b m_{\rm sky}}(\beta)$ by another factor of 2.

4.3. *Example Simulations*

To determine the accuracy and performance of `conviqt`, a detailed comparison with the stable release of the `LevelS totalconvolver` (Wandelt & Gorski 2001; Reinecke et al. 2006) currently compiled on the `planck` cluster at the National Energy Research Science Council (NERSC) was performed. `LevelS` is a simulation package for the generation of time-ordered data (TOD) by the *Planck* satellite (http://www.esa.int/SPECIALS/Planck/index.html). `Totalconvolver` and `conviqt` both calculate a data cube that is fully compatible with `LevelS`. For both codes, data cube is composed of convolved points calculated at a polar angle θ, a longitudinal angle ϕ, and a particular beam orientation (a rotation about the beam axis) ψ. In Section 4.3.1, we compare the output data of both codes in order to detect potential significant discrepancies hinting at code bugs or numerical problems. In Section 4.3.2, we first compare the performance of both codes when run on a single processor. Afterward the scaling behavior of `conviqt` to very high task numbers is demonstrated. We cannot provide a comparison with `totalconvolver` for this experiment, since due to its internal structure it can only run on shared-memory architectures and is therefore limited to a relatively low degree of parallelism.

4.3.1. *Data Cube Comparison*

For $l_{\max} = 2000$, GRASP beams for LFI-19a, $m_{b\max} = 9$, `offset` $= 30$ (see Equation (40)) and a polarized CMB map, there were 153,634,399 points in the data cube. Taking the difference between the `totalconvolver` and `conviqt` data cubes (residual values below), we obtained Table 1 where "avr" refers to the average difference of the two data cubes (`conviqt`(θ,ϕ,ψ) $-$ `totalconvolver`(θ,ϕ,ψ)), σ is the variance of that residual data cube, "Max" refers to the maximum value found in the residual data cube, and "rel" refers to the ratio of σ to the variance of the `totalconvolver` data cube. We see that with `offset` $= 2000$, the two data cubes agree to approximately eight significant digits; for `offset` $= 30$ and 15, they agree to six and four significant digits, respectively.

Table 1
Convolution Output Comparison Between `Conviqt` and `Totalconvolver` for Different `Offsets`

`Offset`	Avr	σ	Max	Rel
2000 (exact)	−1.7e−16	4.1e−13	1.4e−11	4.0e−8
30	−1.2e−16	1.3e−11	5.6e−10	1.3e−6
15	−1.6e−16	3.0e−9	1.1e−7	2.9e−4

Table 2
Comparison in Timing and Memory Consumption Between `Conviqt` and `Totalconvolver` for $l_{\max} = 2000$, $m_{b\max} = 9$, and `Offset` $= 30$

`Code`	Clock (s)	GBytes
`Conviqt`	349	0.37
`Totalconvolver`	1120	0.41

Table 3
Timing and Memory Consumption Comparison Between `Conviqt` and `Totalconvolver`

	`Conviqt`		`Totalconvolver`	
$l_{\max}$	s	GBytes	s	GBytes
256	1.9657	1.51806e−02	11.726	2.32534e−02
512	9.8691	1.76401e−02	54.634	5.00412e−02
768	27.982	2.16856e−02	144.51	9.46579e−02
1024	61.797	2.73190e−02	294.71	1.57092e−01
1280	117.13	3.45383e−02	513.29	2.37350e−01
1536	199.29	4.33445e−02	809.51	3.35554e−01
1792	313.27	5.37376e−02	1264.1	4.51582e−01
2048	621.26	6.57196e−02	1765.2	5.85376e−01

4.3.2. Performance

On a single processor on Jacquard, for $l_{\max} = 2000$ with a GRASP beam (LFI-19a) of $m_{b\max} = 9$, `offset` $= 30$, MC = T including polarization, `conviqt` had the performance shown in Table 2 where "Clock" refers to the time it took to complete the convolution according to the wall clock, while "GBytes" refers to the total memory consumed. These numbers were obtained using NERSC's Integrated Performance Monitoring (IPM) on a 712-CPU Opteron cluster called `Jacquard` running a Linux operating system; each processor on `Jacquard` runs at a clock speed of 2.2 GHz with a theoretical peak performance of 4.4 GFlop s^{-1}. Unlike `totalconvolver`, `conviqt` is a massively parallel code which can be run on machines with distributed memory; running it on a single processor shows that `conviqt` is intrinsically faster and more efficient than `totalconvolver`. The above table is for the case where `offset` $< l_{\max}$, i.e., the case where `conviqt` sacrifices precision for the sake of a speedier convolution.

For the general case where `conviqt` and `totalconvolver` calculate the same thing (`offset`$\rightarrow \infty$), we use the single-code approach to obtain Table 3: these numbers are plotted in Figure 4. The top plot shows that `conviqt` is considerably faster than `totalconvolver` for $l_{\max} < 2048$; however, because both codes scale as $l^3_{\max}$ as $l_{\max} \rightarrow \infty$, the gap between their total wall clock times will narrow. It is also seen that `conviqt` consumes significantly less memory.

The scaling of `conviqt` timings as a function of the total number of processors is very good as can be seen from Table 4 and the plot in Figure 5. For $l_{\max} = 4096$ and $m_b = 14$ with polarized beam and sky, the plot in Figure 5 shows an inverse relationship between the timings t and the number of processors n

$$n = t_1/(t - t_0), \tag{46}$$

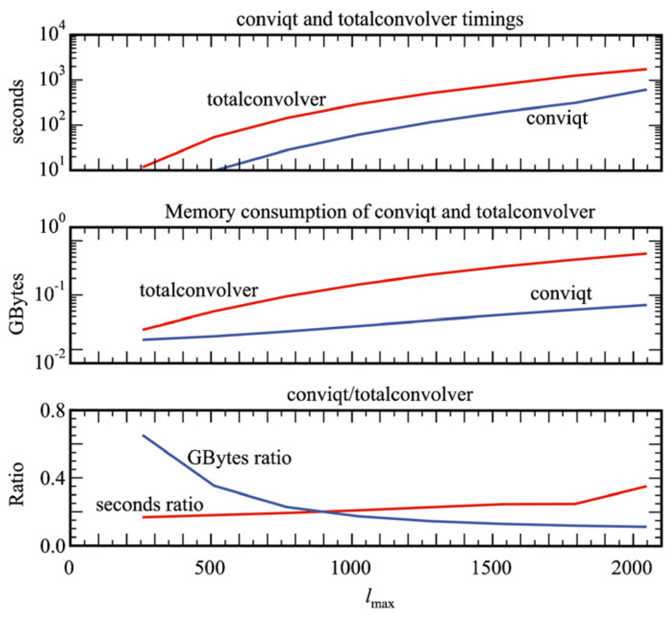

Figure 4. Top plot compares the wall clock performance of `conviqt` and `totalconvolver` at $l_{\max}$ intervals of 256 starting at $l_{\max} = 256$ and the middle plot compares the memory needs in GBytes of the two codes. The lower plot shows the ratio of wall clock and GBytes of the two codes.

(A color version of this figure is available in the online journal.)

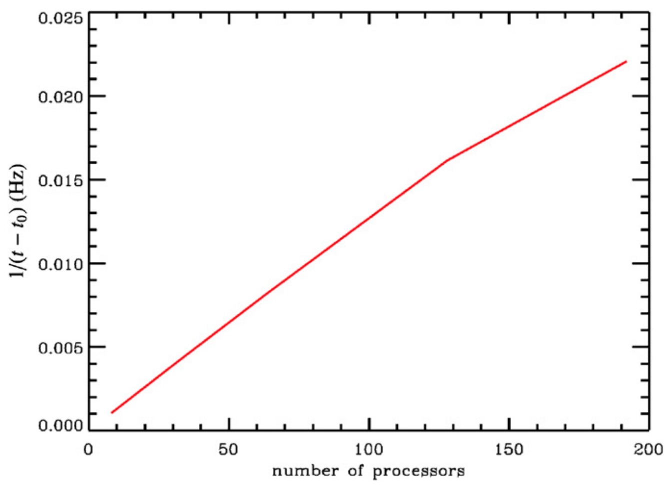

Figure 5. `Timings of conviqt as the number of processors is increased with` $l_{\max} = 4096$ `and` $m_{b\max} = 14$.

(A color version of this figure is available in the online journal.)

Table 4
`Conviqt` Scaling with the Number of Processors, with $t_0 = 7.15$ s

Number of Processors	Seconds	$1/(t - t_0)$ (Hz)
8	965.46	0.0010435037
16	486.29	0.0020870727
32	247.83	0.0041548945
64	128.43	0.0082453826
128	69.06	0.016152479
192	52.47	0.022065313

where $t_0 = 7.15$ s and $t_1 = 7666.72$ s are parameters that can be thought of, respectively, as the time needed for the intrinsically scalar part of the code and the time needed to perform the convolution on a single processor if $t_0 = 0$. This relationship is satisfied up to a convolution distributed on 128 processors. This plot was obtained using the single-code approach run on the NERSC cluster called `Planck`, a 256 cores cluster of Opteron 2350 2.0 GHz processors. To measure the scaling behavior of `conviqt`, no output file was created to avoid skewing the scaling law with the time it takes to write the file (tens of seconds for a 4 GB file). As the number of processors increases and the time required to perform the convolution diminishes to less than a minute, the timings become dominated with operations that have nothing to do with the convolution; among these are the reading of the input data sets (which are read in full by all MPI tasks), the interprocess communication and various calculations which are performed redundantly on all tasks, because communicating the results would be more expensive. Increasing the number of tasks (while keeping the problem size constant) also means a smaller number of β angles per task, which decreases the achievable quality of load balancing. In addition, different runs with identical inputs show variations of a few seconds in wall clock timings that have an increasing relative impact on the decreasing timings stemming from using larger numbers of processors; the most likely explanation for this is differences in the exact nature of process startup and disk access, which is not exactly reproducible in this kind of computing environment.

5. SUMMARY

New algorithms for the efficient and accurate calculation of Wigner matrix elements were presented. These algorithms were used in a full sky convolution, massively parallel algorithm called `conviqt` that was shown to be significantly more efficient and much faster than the only other algorithm currently available.

G.P. thanks Maura Sandri for the GRASP 8 beams used in these simulations and Charles Lawrence for useful comments on this manuscript. We gratefully acknowledge support by the NASA Science Mission Directorate via the US Planck Project. The research described in this paper was partially carried out at the Jet propulsion Laboratory, California Institute of Technology, under a contract with NASA. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract DE-AC02-05CH11231. M.R. is supported by the German Aeronautics Center and Space Agency (DLR), under program 50-OP-0901, funded by the Federal Ministry of Economics and Technology.

APPENDIX: SYMMETRIES OF $d^l_{mm'}(\beta)$

$$d^l_{mm'}(\beta) = (-1)^{m-m'} d^l_{-m-m'}(\beta), \tag{A1}$$

$$d^l_{mm'}(\beta) = (-1)^{m-m'} d^l_{m'm}(\beta), \tag{A2}$$

$$d^l_{mm'}(\beta) = d^l_{-m'-m}(\beta), \tag{A3}$$

$$d^l_{mm'}(-\beta) = d^l_{m'm}(\beta), \tag{A4}$$

$$d^l_{mm'}(-\beta) = (-1)^{m-m'} d^l_{mm'}(\beta), \tag{A5}$$

$$d^l_{mm'}(\pi - \beta) = (-1)^{l-m'} d^l_{-mm'}(\beta), \tag{A6}$$

$$d^l_{mm'}(\pi - \beta) = (-1)^{l+m'} d^l_{m-m'}(\beta). \tag{A7}$$

REFERENCES

Edmonds, A. R. 1957, Angular Momentum in Quantum Mechanics (Princeton, NJ: Princeton Univ. Press)
Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. 1988, Numerical Recipes in C++ (Cambridge: Cambridge Univ. Press)
Reinecke, M., Dolag, K., Hell, R., Bartelmann, M., & Ensslin, T. 2006, A&A, 445, 373
Risbo, T. 1996, J. Geod., 70, 383
Rowe, D. J., de Guise, H., & Sanders, B. C. 2001, J. Math. Phys., 42, 2315
Varshalovich, D. A., Moskalev, A. N., & Khersonskii, V. K. 1988, Quantum Theory of Angular Momentum (Singapore: World Scientific)
Wandelt, B. D., & Gorski, K. M. 2001, Phys. Rev. D., 63, 123002