Max-Planck-Institut
für Mathematik
in den Naturwissenschaften
Leipzig

Fast and Exact Projected Convolution for Non-equidistant Grids - Extended Version

(revised version: March 2007)

by

Wolfgang Hackbusch

Preprint no.: 102 2006

Fast and Exact Projected Convolution for Non-equidistant Grids Extended Version

Wolfgang Hackbusch
Max-Planck-Institut *Mathematik in den Naturwissenschaften*
Inselstr. 22, D-04103 Leipzig

Abstract

Usually, the fast evaluation of a convolution integral $\int_{\mathbb{R}} f(y)g(x-y)\mathrm{d}y$ requires that the functions f, g are discretised on an equidistant grid in order to apply the fast Fourier transform. Here we discuss the efficient performance of the convolution in locally refined grids. More precisely, the convolution result is projected into some given locally refined grid. Under certain conditions, the overall costs are still $\mathcal{O}(N \log N)$, where N is the sum of the dimensions of the subspaces containing f, g and the resulting function.

AMS Subject Classifications: 44A35, 42A55
Key words: convolution integral, non-uniform grids, discrete convolution

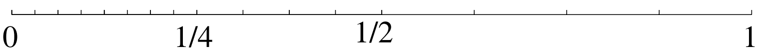
1 Introduction

We consider the convolution integral

$$\omega_{\text{exact}}(x) := (f * g)(x) := \int_{\mathbb{R}} f(y)g(x-y)\mathrm{d}y \tag{1.1}$$

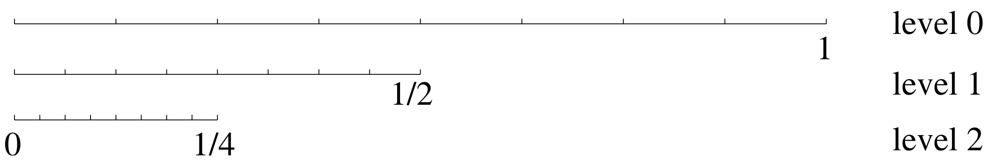
for functions f, g of bounded support and note that $f * g = g * f$.

The computations are restricted to f, g which are piecewise polynomials defined on *locally refined meshes* with possible discontinuities at the grid points. A simple example of such a locally refined mesh is depicted below:



(1.2a)

The mesh size 1/8 in $[1/2, 1]$, 1/16 in $[1/4, 1/2]$ and 1/16 in $[0, 1/4]$ is a typical refinement towards $x = 0$. The depicted mesh can be decomposed into different levels as indicated in



(1.2b)

The latter representation uses several levels. Each level ℓ is associated with an *equidistant* grid of size

$$h_\ell := 2^{-\ell}h \qquad (0 \leq \ell \leq L) \tag{1.3}$$

(in the above figure with $h = 1/8$). The largest level number appearing in the grid hierarchy will be denoted by L.

Such a refinement approach is well-known from the adaptive wavelet technique. Grids like the depicted ones are obtainable from a first coarse grid with mesh size h at level $\ell = 0$ after a recursive local refinement by halving certain subintervals. The exact description of the locally refined mesh will be given in Section 2.

The standard tool for convolutions is the Fast Fourier Transform (FFT), which, however, applies only for data in a uniform grid. In principle, one can convert the functions f, g from the refined grid into functions defined on the finest uniform grid of level L (with step size h_L). Since this step increases the data size extremely, the almost linear complexity of FFT does not help.

Remark 1.1 *Another construction of graded meshes is based on a grid* $\left\{\theta(\frac{i}{n}) : i_1 \leq i \leq i_2\right\}$*, where* θ *is a smooth monotonous mapping. This corresponds to the substitution* $y = \theta(\eta)$ *in* $f(\theta(\eta)) =: F(\eta)$ *which is now given on an* equidistant *grid. However, the integral* $(f * g)(x) = \int_{\mathbb{R}} f(y)g(x-y)\mathrm{d}y = \int_{\mathbb{R}} F(\eta)g(x-\theta(\eta))\theta'(\eta)\mathrm{d}\eta$ *is not a convolution of* F *with another function.*

In principle, one can approximate $f * g$ via Fourier transform: Also for non-equidistant grids there are ways to approximate the Fourier transform $\hat{f}$ and $\hat{g}$ (see [4]) and the back transform of $\hat{f} \cdot \hat{g}$ would yield an approximation of $f * g$. A fast algorithm for a generalised convolution is described in [5]. However, in these approaches the approximation error depends on the interpolation error and we do not guarantee any smoothness of f or g. In contrary, the use of locally refined meshes indicates a nonsmooth situation. In our approach we avoid interpolation errors, since the quantities of interest are computed exactly.

To cover the general case, we will allow that the functions f and g involved in the convolution belong to different locally refined meshes.

Remark 1.2 *The convolution* $\omega_{\mathrm{exact}} = f * g$ *leads to two difficulties.*

a) First, the convolution of piecewise constant functions is no longer piecewise constant but piecewise linear and globally continuous[1]*. Hence, the image belongs to another class of functions.*

b) The second difficulty is even more disturbing. For efficiency it is an important fact that f *and* g *are given on locally refined meshes (instead of an equidistant grid of the smallest size). As mentioned before, the convolution is piecewise linear, but the intervals in which it is linear correspond to the* smallest *step size appearing in the representations of* f *and* g*. Therefore,* ω_{exact} *not only belongs to another function class, its exact description requires also much more data than* f *and* g*.*

[1] If, more general, f and g are piecewise polynomials of the respective degrees p and q, the convolution is a piecewise polynomial of degree $p + q + 1$.

However, since the locally refined meshes have the purpose to approximate some functions f_{exact} and g_{exact} in an adaptive way by f and g, it is only natural to approximate ω_{exact} also by a simpler function ω in a third given locally refined mesh, e.g., again by piecewise constant functions. We use the L^2-orthogonal projection onto this space to obtain ω from ω_{exact}. The result ω can be considered as the *Galerkin approximation* of the convolution operation.

Therefore, the goal of the algorithm is to compute ω as the L^2-orthogonal projection of $f * g$. Note that we compute the *exact* L^2-orthogonal projection, i.e., there is no approximation error except the unavoidable projection error.

We return to the situation of Remark 1.1. If f and g are given via a grid of the form $\left\{\theta(\frac{i}{n}) : i_1 \leq i \leq i_2\right\}$, they can be projected into suitable locally refined grids as requested for our approach. The results $\tilde{f}$ and $\tilde{g}$ yield the projected convolution ω which returned onto the desired graded mesh via projection. If the grids are chosen accordingly to the smoothness of the functions f, g, ω, the additional projection error is not larger than the already involved approximation error.

Section 2 gives a precise definition of locally refined meshes and of the corresponding ansatz spaces $\mathcal{S}^f, \mathcal{S}^g$ which f and g belong to and of the target space $\mathcal{S}^\omega$ for the projection ω of $\omega_{\text{exact}} = f * g$. In particular, the basis functions Φ_i^ℓ are introduced in §2.3. From §2.3 to §5 we restrict the description to the case of piecewise constant functions.

Section 4 introduces several families of coefficients which are essential for the representation of the projected values. Some of these coefficients will also appear later in the algorithm.

The main chapter of this paper is Section 5 which describes the algorithm. Three disjoint cases A, B, C must be treated differently (see §§5.1-5.3).

In all three cases mentioned above, we have to perform a discrete convolution of sequences by means of the FFT technique. Section 6 is devoted to this problem. It starts with the definition of the notations N_d and N_c for the data size (§6.1). The well-known basic FFT algorithm for the discrete convolution (§6.2) is modified in §§6.3, 6.5, 6.6 to minimise the computational work.

The cost of the previous algorithm is analysed in Section 7. There, under certain conditions, the bound

$$\mathcal{O}\left(N \log N\right)$$

is derived, where N describes the data size of the factors f, g and the projected convolution $\omega = P(f * g)$.

So far, the case of piecewise constant functions is considered. In Section 8, the generalisation to arbitrary piecewise polynomial approximation spaces is discussed.

A variant of the convolution (1.1) is the convolution of *periodic* functions. Their computations follows the same lines (see Section 9).

The Appendix contains alternative algorithms, which however turn out to be less efficient than the algorithms presented in the Section 5.

2 Spaces

2.1 The Locally Refined Meshes

The grids depicted in (1.2b) are embedded into infinite grids $\mathcal{M}_\ell$ which are defined below. With h_ℓ from (1.3) we denote the subintervals of level ℓ by

$$I_\nu^\ell := [\nu h_\ell, (\nu+1)\, h_\ell) \qquad \text{for } \nu \in \mathbb{Z},\ \ell \in \mathbb{N}_0. \tag{2.1}$$

This defines the meshes

$$\mathcal{M}_\ell := \left\{ I_\nu^\ell : \nu \in \mathbb{Z} \right\} \qquad \text{for } \ell \in \mathbb{N}_0. \tag{2.2}$$

The (equidistant) grid points of $\mathcal{M}_\ell$ are $\{\nu h_\ell : \nu \in \mathbb{Z}\}$.

A finite and *locally refined mesh* $\mathcal{M}$ is a set of finitely many disjoint intervals from various levels, i.e.,[2]

$$\mathcal{M} \subset \bigcup\nolimits_{\ell\in\mathbb{N}_0} \mathcal{M}_\ell, \qquad \text{all } I, I' \in \mathcal{M} \text{ with } I \neq I' \text{ are disjoint}, \qquad \#\mathcal{M} < \infty\,. \tag{2.3}$$

Definition (2.3) corresponds to the representation (1.2a), whereas (1.2b) gives rise to the following dynamic definition. Let $\mathcal{M}_0' \subset \mathcal{M}_0$ be a finite part of the infinite grid $\mathcal{M}_0$. Certain intervals $I_\nu^0 \in \mathcal{M}_0'$ are refined, i.e., I_ν^0 is removed from $\mathcal{M}_0'$ and the resulting new subintervals $I_{2\nu}^1$ and $I_{2\nu+1}^1$ of halved size are added to $\mathcal{M}_1' \subset \mathcal{M}_1$ (initialised by $\mathcal{M}_1' := \emptyset$). Recursively, certain $I_\nu^\ell \in \mathcal{M}_\ell'$ are replaced by $I_{2\nu}^{\ell+1}, I_{2\nu+1}^{\ell+1} \in \mathcal{M}_{\ell+1}'$. Let $\mathcal{M}_\ell''$ be the final value of $\mathcal{M}_\ell'$ after terminating the refinement process. Then $\mathcal{M} := \bigcup_{\ell\in\mathbb{N}_0} \mathcal{M}_\ell''$ yields the set from (2.3).

2.2 The Ansatz Spaces

For each interval $I = I_\nu^\ell \in \mathcal{M}$ we can fix a polynomial degree $p_\nu^\ell \in \mathbb{N}_0$. Then a piecewise polynomial space $\mathcal{S}$ corresponding to the mesh $\mathcal{M}$ and the polynomial degree distribution $\left\{p_\nu^\ell\right\}$ is defined by

$$\mathcal{S} = \mathcal{S}\left(\mathcal{M}\right) = \left\{ \begin{array}{c} \phi \in L^\infty(\mathbb{R}) : \phi|_{I_\nu^\ell} \text{ polynomial of degree } \leq p_\nu^\ell \text{ if } I_\nu^\ell \in \mathcal{M}, \\ \phi(x) = 0 \text{ if } x \notin I \text{ for all } I \in \mathcal{M} \end{array} \right\}.$$

Such a space $\mathcal{S}$ is a typical result of the *hp*-adaptive refinement procedure.

The two factors f, g of the convolution as well as the (projected) image ω may be organised by three different locally refined meshes

$$\mathcal{M}^f, \quad \mathcal{M}^g, \quad \mathcal{M}^\omega, \tag{2.4}$$

which are all of the form (2.3). These give rise to three spaces

$$\mathcal{S}^f := \mathcal{S}\left(\mathcal{M}^f\right), \quad \mathcal{S}^g := \mathcal{S}\left(\mathcal{M}^g\right), \quad \mathcal{S}^\omega := \mathcal{S}\left(\mathcal{M}^\omega\right).$$

We recall that $f \in \mathcal{S}^f$ and $g \in \mathcal{S}^g$ are the input data, while the result is the exact L^2-orthogonal projection of $f * g$ onto $\mathcal{S}^\omega$.

In [2], the underlying problem takes the principal form $\mathrm{d}f/\mathrm{d}t = \ldots + f * f$ (here $\mathcal{S}^f = \mathcal{S}^g = \mathcal{S}^\omega$ is the appropriate choice of the spaces). In [1], the problem to be solved is a fixed point equation $f = \ldots + f * g$. Therefore $\mathcal{S}^f = \mathcal{S}^\omega$ holds, while $\mathcal{S}^g$ may be chosen differently.

[2] The sign # denotes the cardinality of a set.

2.3 Basis Functions

Functions from $\mathcal{S}(\mathcal{M})$ may be discontinuous at the grid points of the mesh. This fact has the advantage that the basis functions spanning $\mathcal{S}(\mathcal{M})$ have minimal support (the support is just one interval of $\mathcal{M}$). For the case $p_\nu^\ell > 0$ of piecewise non-constant functions, the basis functions will be described in Section 8.

Here, we consider the case of piecewise constant functions, i.e., $p_\nu^\ell = 0$. The generating function is

$$\Phi_0^0(x) := \begin{cases} 1/\sqrt{h} & \text{if } x \in (0,h)\,, \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

leading to the basis functions of level $\ell = 0$:

$$\Phi_i^0(x) := \Phi_0^0(x - ih) \qquad (i \in \mathbb{Z})\,.$$

For levels $\ell > 0$ we have

$$\Phi_0^\ell(x) := 2^{\ell/2}\Phi_0^0(2^\ell x), \qquad \Phi_i^\ell(x) := \Phi_0^\ell(x - ih_\ell) = 2^{\ell/2}\Phi_i^0(2^\ell x) \qquad (\ell \in \mathbb{N}_0,\ i \in \mathbb{Z})\,. \tag{2.6}$$

Note that $\mathrm{supp}(\Phi_i^\ell) = I_i^\ell$. The space $\mathcal{S}(\mathcal{M})$ has the representation

$$\mathcal{S}(\mathcal{M}) = \mathrm{span}\left\{\Phi_i^\ell : I_i^\ell \in \mathcal{M}\right\}.$$

In the following, we make use of relations which are well-known from wavelet approaches. Let $\mathcal{S}_\ell$ be the space of piecewise constant functions of level ℓ (on the infinite mesh $\mathcal{M}_\ell$ from (2.2)):

$$\mathcal{S}_\ell := \mathrm{span}\left\{\Phi_i^\ell : i \in \mathbb{Z}\right\} \qquad (\ell \in \mathbb{N}_0)\,. \tag{2.7}$$

For fixed ℓ, the basis $\left\{\Phi_i^\ell : i \in \mathbb{Z}\right\}$ is orthonormal due to the chosen scaling.

The spaces are nested, i.e.,

$$\mathcal{S}_\ell \subset \mathcal{S}_{\ell+1}.$$

In particular, Φ_i^ℓ can be represented by means of $\Phi_j^{\ell+1}$:

$$\Phi_i^\ell = \frac{1}{\sqrt{2}}\left(\Phi_{2i}^{\ell+1} + \Phi_{2i+1}^{\ell+1}\right). \tag{2.8}$$

3 Notations and Definition of the Problem

3.1 Representations of $f \in \mathcal{S}^f$ and $g \in \mathcal{S}^g$

Following the definition of $\mathcal{S}^f$, we have $\mathcal{S}^f = \mathrm{span}\left\{\Phi_i^\ell : I_i^\ell \in \mathcal{M}^f\right\}$. We can decompose the set $\mathcal{M}^f$ into different levels: $\mathcal{M}^f = \bigcup_{\ell=0}^{Lf} \mathcal{M}_\ell^f$, where $\mathcal{M}_\ell^f := \mathcal{M}^f \cap \mathcal{M}_\ell$. This gives rise to the related index set

$$\mathcal{I}_\ell^f := \left\{i \in \mathbb{Z} : I_i^\ell \in \mathcal{M}_\ell^f\right\} \tag{3.1}$$

and to the corresponding decomposition

$$\mathcal{S}^f = \bigcup\nolimits_{\ell=0}^{L^f} \mathcal{S}_\ell^f \qquad \text{with } \mathcal{S}_\ell^f = \mathrm{span}\left\{\Phi_i^\ell : i \in \mathcal{I}_\ell^f\right\}.$$

Here, L^f is the largest level ℓ with $\mathcal{M}_\ell^f \neq \emptyset$.

We start from the representation

$$f = \sum_{\ell=0}^{L^f} f_\ell, \qquad f_\ell = \sum_{i\in\mathcal{I}_\ell^f} f_i^\ell \Phi_i^\ell \in \mathcal{S}_\ell^f\,, \tag{3.2}$$

and, similarly,

$$g = \sum_{\ell=0}^{L^g} g_\ell, \qquad g_\ell = \sum_{i\in\mathcal{I}_\ell^g} g_i^\ell \Phi_i^\ell \in \mathcal{S}_\ell^g\,, \tag{3.3}$$

for the factors f, g of the convolution.

3.2 Projection P_ℓ

The L^2-orthogonal projection P_ℓ onto $\mathcal{S}_\ell$ from (2.7) is defined by

$$P_\ell\,\varphi := \sum_{i\in\mathbb{Z}} \left\langle \varphi, \Phi_i^\ell \right\rangle \Phi_i^\ell \tag{3.4}$$

with $\langle\varphi,\psi\rangle = \int_{\mathbb{R}} \varphi\psi \mathrm{d}x$, provided that $\left\{\Phi_i^\ell\right\}$ forms an orthonormal basis[3] as it holds for Φ_i^ℓ from (2.6).

3.3 Definition of the Basic Problem

We use the decomposition into scales expressed by $f = \sum_{\ell'=0}^{L^f} f_{\ell'}$ and $g = \sum_{\ell=0}^{L^g} g_\ell$ (see (3.2) and (3.3)). The convolution $f * g$ can be written as

$$f * g = \sum_{\ell'=0}^{L^f}\sum_{\ell=0}^{L^g} f_{\ell'} * g_\ell\,.$$

Since the convolution is symmetric, we can rewrite the sum as

$$f * g = \sum_{\ell'\leq\ell} f_{\ell'} * g_\ell + \sum_{\ell<\ell'} g_\ell * f_{\ell'}\,, \tag{3.5}$$

where ℓ', ℓ are restricted to the level intervals $0 \leq \ell' \leq L^f$, $0 \leq \ell \leq L^g$. Hence, the basic task is as follows.

Problem 3.1 *Let $\ell' \leq \ell$, $f_{\ell'} \in \mathcal{S}_{\ell'}$, $g_\ell \in \mathcal{S}_\ell$, and $\ell'' \in \mathbb{N}_0$ a further level. Then, the projection $P_{\ell''}(f_{\ell'} * g_\ell)$ is to be computed. More precisely, only the restriction of $P_{\ell''}(f_{\ell'} * g_\ell)$ to $\bigcup_{i\in\mathcal{I}_{\ell''}^\omega} I_i^{\ell''}$ is needed, since only this part appears in $\mathcal{S}_{\ell''}^\omega$.*

Because of the splitting (3.5), we may assume $\ell' \leq \ell$ without loss of generality. In the case of the second sum one has to interchange the rôles of the symbols f and g.

Before we present the solution algorithm in Section 5, we introduce some further notations in the next Section.

[3]Otherwise, a Gram matrix has to be inverted.

4 Auxiliary Coefficients

4.1 γ-Coefficients

For level numbers $\ell'', \ell', \ell \in \mathbb{N}_0$ and all $i, j, k \in \mathbb{Z}$ we define

$$\gamma_{i,j,k}^{\ell'',\ell',\ell} := \iint \Phi_i^{\ell''}(x)\, \Phi_j^{\ell'}(y)\, \Phi_k^{\ell}(x-y)\mathrm{d}x\mathrm{d}y \tag{4.1}$$

(all integrations over $\mathbb{R}$). We remark that $\gamma_{i,j,k}^{\ell'',\ell',\ell} = \left\langle \Phi_i^{\ell''}, \Phi_j^{\ell'} * \Phi_k^{\ell} \right\rangle$ is the L^2-scalar product of the basis function $\Phi_i^{\ell''}$ and the convolution $\Phi_j^{\ell'} * \Phi_k^{\ell}$.

The connection to the computation of the projection

$$\omega_{\ell''} = P_{\ell''}\left(f_{\ell'} * g_{\ell}\right) \tag{4.2}$$

of the convolution $f_{\ell'} * g_{\ell}$ from Problem 3.1 is as follows. $\omega_{\ell''}$ is a function represented by $\omega_{\ell''} = \sum_{i\in\mathbb{Z}}^{\ell} \omega_i^{\ell''} \Phi_i^{\ell''}$, where the coefficients $\omega_i^{\ell''}$ result from

$$\begin{aligned} \omega_i^{\ell''} &= \int (f_{\ell'} * g_{\ell})(x)\, \Phi_i^{\ell''}(x)\, \mathrm{d}x = \int \Phi_i^{\ell''}(x) \left(\sum_{j\in\mathbb{Z}} f_j^{\ell'} \Phi_j^{\ell'} * \sum_{k\in\mathbb{Z}} g_k^{\ell} \Phi_k^{\ell} \right)(x)\, \mathrm{d}x \\ &= \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \iint \Phi_i^{\ell''}(x)\Phi_j^{\ell'}(y)\Phi_k^{\ell}(x-y)\, \mathrm{d}x\mathrm{d}y = \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell}\, \gamma_{i,j,k}^{\ell'',\ell',\ell} \end{aligned} \tag{4.3}$$

(again, orthonormality of the basis $\{\Phi_i^{\ell''}\}$ is used).

The recursion formula (2.8) can be applied to all three basis functions in the integrand $\Phi_i^{\ell''}(x)\Phi_j^{\ell'}(y)\Phi_k^{\ell}(x-y)$ of $\gamma_{i,j,k}^{\ell'',\ell',\ell}$. The resulting formulae for $\gamma_{i,j,k}^{\ell'',\ell',\ell}$ are given in the next Remark.

Remark 4.1 *For all $\ell'', \ell', \ell \in \mathbb{N}_0$ and all $i, j, k \in \mathbb{Z}$ we have*

$$\gamma_{i,j,k}^{\ell'',\ell',\ell} = \frac{1}{\sqrt{2}} \left(\gamma_{2i,j,k}^{\ell''+1,\ell',\ell} + \gamma_{2i+1,j,k}^{\ell''+1,\ell',\ell} \right) \tag{4.4a}$$

$$= \frac{1}{\sqrt{2}} \left(\gamma_{i,2j,k}^{\ell'',\ell'+1,\ell} + \gamma_{i,2j+1,k}^{\ell'',\ell'+1,\ell} \right) \tag{4.4b}$$

$$= \frac{1}{\sqrt{2}} \left(\gamma_{i,j,2k}^{\ell'',\ell',\ell+1} + \gamma_{i,j,2k+1}^{\ell'',\ell',\ell+1} \right). \tag{4.4c}$$

4.2 Simplified γ-Coefficients

For levels ℓ, ℓ', ℓ'' with $\ell \geq \max\{\ell', \ell''\}$ we set

$$\gamma_{\nu}^{\ell'',\ell',\ell} := \iint \Phi_0^{\ell''}(x)\Phi_0^{\ell'}(y)\Phi_{\nu}^{\ell}(x-y)\mathrm{d}x\mathrm{d}y \qquad (\nu \in \mathbb{Z}), \tag{4.5}$$

i.e., $\gamma_{\nu}^{\ell'',\ell',\ell} = \gamma_{0,0,\nu}^{\ell'',\ell',\ell} = \left\langle \Phi_0^{\ell''}, \Phi_0^{\ell'} * \Phi_{\nu}^{\ell} \right\rangle$. We call these coefficients simplified γ-coefficients, since only one subindex ν is involved.

Under the condition $\ell \geq \max\{\ell', \ell''\}$, it suffices to use the quantities $\gamma_{\nu}^{\ell',\ell'',\ell}$ from (4.5) as shown in the next Lemma.

Lemma 4.2 *Let $\ell \geq \max\{\ell', \ell''\}$. Then*

$$\gamma_{i,j,k}^{\ell'',\ell',\ell} = \gamma_{k-i2^{\ell-\ell''}+j2^{\ell-\ell'}}^{\ell'',\ell',\ell} \qquad \text{for any } i,j,k \in \mathbb{Z}. \tag{4.6}$$

Proof. Using (2.6), we get by substitution

$$\begin{aligned}
\gamma_{i,j,\nu}^{\ell'',\ell',\ell} &= \iint \Phi_i^{\ell''}(x)\Phi_j^{\ell'}(y)\Phi_\nu^{\ell}(x-y)\mathrm{d}x\mathrm{d}y = \iint \Phi_0^{\ell''}(x-ih_{\ell''})\Phi_0^{\ell'}(y-jh_{\ell'})\Phi_k^{\ell}(x-y)\mathrm{d}x\mathrm{d}y \\
&= \iint \Phi_0^{\ell''}(x)\Phi_0^{\ell'}(y)\Phi_k^{\ell}(x-y+ih_{\ell''}-jh_{\ell'})\mathrm{d}x\mathrm{d}y \\
&= \iint \Phi_0^{\ell''}(x)\Phi_0^{\ell'}(y)\Phi_k^{\ell}(x-y+i2^{\ell-\ell''}h_\ell - j2^{\ell-\ell'}h_\ell)\mathrm{d}x\mathrm{d}y \\
&= \iint \Phi_0^{\ell''}(x)\Phi_0^{\ell'}(y)\Phi_{k-i2^{\ell-\ell''}+j2^{\ell-\ell'}}^{\ell}(x-y)\mathrm{d}x\mathrm{d}y = \gamma_{k-i2^{\ell-\ell'}+j2^{\ell-\ell''}}^{\ell'',\ell',\ell}.
\end{aligned}$$

■

Remark 4.3 *In the case of piecewise constant functions, i.e. (2.5), the values of $\gamma_\nu^{\ell,\ell,\ell}$ are*

$$\gamma_0^{\ell,\ell,\ell} = \gamma_{-1}^{\ell,\ell,\ell} = \sqrt{h_\ell}/2 \quad \text{and} \quad \gamma_\nu^{\ell,\ell,\ell} = 0 \text{ for } \nu \notin \{-1,0\}.$$

4.3 G- and Γ-Coefficients

As stated in (4.3), we have to compute $\sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \gamma_{i,j,k}^{\ell'',\ell',\ell}$. Performing only the sum over k, leads us to

$$G_{i,j}^{\ell'',\ell',\ell} := \sum_{k\in\mathbb{Z}} g_k^{\ell}\, \gamma_{i,j,k}^{\ell'',\ell',\ell}. \tag{4.7}$$

We can introduce simpler G-coefficients, where the first or second subindex is fixed by zero.

Remark 4.4 *In the case $\ell'' \leq \ell' \leq \ell$ there holds*

$$G_{i,j}^{\ell'',\ell',\ell} = \sum_{k\in\mathbb{Z}} g_k^{\ell}\, \gamma_{k-i2^{\ell-\ell''}+j2^{\ell-\ell'}}^{\ell'',\ell',\ell} = G_{0,j-i2^{\ell'-\ell''}}^{\ell'',\ell',\ell}, \tag{4.8}$$

whereas for $\ell' \leq \ell'' \leq \ell$

$$G_{i,j}^{\ell'',\ell',\ell} = \sum_{k\in\mathbb{Z}} g_k^{\ell}\, \gamma_{k-i2^{\ell-\ell''}+j2^{\ell-\ell'}}^{\ell'',\ell',\ell} = G_{i-j2^{\ell''-\ell'},0}^{\ell'',\ell',\ell}. \tag{4.9}$$

Using the recursions (4.4a,b) from Remark 4.1, one proves the following result.

Remark 4.5 *For all $\ell'', \ell', \ell \in \mathbb{N}_0$ and all $i,j \in \mathbb{Z}$ we have*

$$\begin{aligned}
G_{i,j}^{\ell'',\ell',\ell} &= \frac{1}{\sqrt{2}}\left(G_{2i,j}^{\ell''+1,\ell',\ell} + G_{2i+1,j}^{\ell''+1,\ell',\ell}\right) && \text{(4.10a)} \\
&= \frac{1}{\sqrt{2}}\left(G_{i,2j}^{\ell'',\ell'+1,\ell} + G_{i,2j+1}^{\ell'',\ell'+1,\ell}\right). && \text{(4.10b)}
\end{aligned}$$

If the first two levels are equal: $\ell'' = \ell' \leq \ell$, the coefficients are denoted by

$$\Gamma_i^{\ell',\ell} := G_{i,0}^{\ell',\ell',\ell} = G_{0,-i}^{\ell',\ell',\ell} = \sum_{k\in\mathbb{Z}} g_k^{\ell}\, \gamma_{k-i2^{\ell-\ell'}}^{\ell',\ell',\ell}. \tag{4.11}$$

5 Algorithm

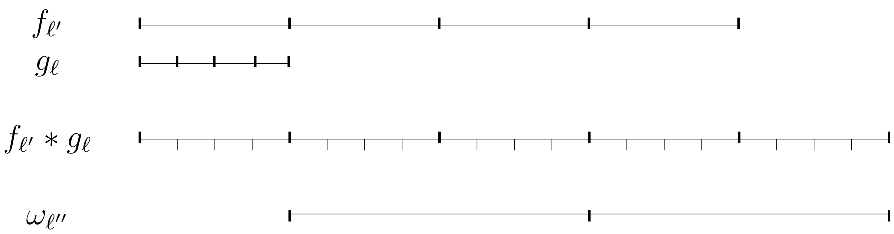
A general assumption of this Section is the choice of piecewise constant functions.

In Problem 3.1 three level numbers ℓ'', ℓ', ℓ appear. Without loss of generality $\ell' \leq \ell$ holds. In the following we have to distinguish the following three cases:

$$\begin{array}{lccccc} (A) & \ell'' & \leq & \ell' & \leq & \ell, \\ (B) & \ell' & < & \ell'' & \leq & \ell, \\ (C) & \ell' & \leq & \ell & < & \ell''. \end{array} \tag{5.1}$$

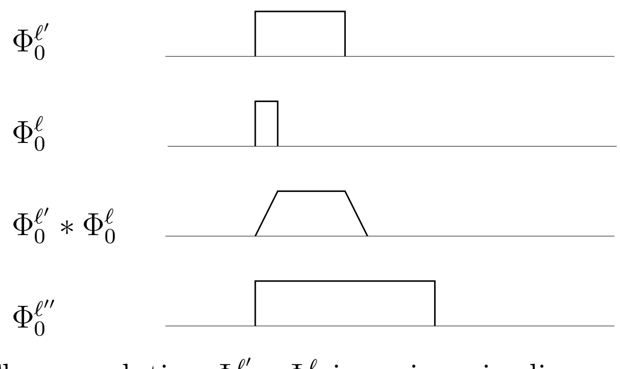
5.1 Case A: $\ell'' \leq \ell' \leq \ell$

Case A is illustrated by the following figure.



In the figure, the difference $\ell - \ell' = 2$ corresponds to the fact that g_ℓ is given on a grid of step size $h_\ell = h_{\ell'}/4$. The given intervals should show the support of the functions $f_{\ell'}$ and g_ℓ. The convolution $f_{\ell'} * g_\ell$ is a piecewise linear function, where the pieces correspond to the smaller step size h_ℓ. The projection $P_{\ell''}$ of $f_{\ell'} * g_\ell$ is required in two intervals. Because of $\ell'' \leq \ell' \leq \ell$ the step size $h_{\ell''}$ is equal or larger than the other ones. In the figure, $\ell'' = \ell' - 1$ is chosen. Note that in each interval of level ℓ'' the function $\omega_{\ell''}$ is a certain average of 8 pieces of $f_{\ell'} * g_\ell$.

Another illustration of Case A can be given by means of the related basis functions[4].



The convolution $\Phi_0^{\ell'} * \Phi_0^{\ell}$ is a piecewise linear function. The projection $P_{\ell''}$ uses the scalar products of shifts of the basis function $\Phi_0^{\ell''}$ shown in the last row.

The following algorithm has to compute the projection of $\omega_{\ell''} = P_{\ell''}\omega_{\text{exact}}$ of $\omega_{\text{exact}} := f_{\ell'} * g_\ell$. A straightforward but naive approach would be to compute ω_{exact} first and then its projection. The problem is that in the case $\ell' \ll \ell$, the product $f_{\ell'} * g_\ell$ requires $\mathcal{O}(2^{\ell-\ell'} N_c(f_{\ell'}) + N_c(g_\ell))$ data (the data size $N_c(\cdot)$ is introduced in §6.1). The possibly large factor $2^{\ell-\ell'}$ underlines the difficulty described in item b) of Remark 1.2 and spoils efficiency.

[4]Note that the heights of the basis functions $\Phi_0^{\ell'}$, Φ_0^{ℓ} are not correctly reproduced in the figure.

The projection would map the many data into few: each component $\omega_i^{\ell''}$ of $\omega_{\ell''}$ is an average of $2^{\ell-\ell''}$ data of ω_{exact}. The essence of the following algorithm is to incorporate the projection $P_{\ell'}$ before a discrete convolution is performed.

5.1.1 Computation of Γ-Coefficients (Step 1)

Step 1a We start with the sequence $\Gamma_{\ell,\ell} = (\Gamma_i^{\ell,\ell})_{i\in\mathbb{Z}}$. Formally, $\Gamma_{\ell,\ell}$ is a kind of discrete convolution $\sum_{k\in\mathbb{Z}} g_k^\ell \gamma_{k-i}^{\ell,\ell,\ell}$ of the sequences $(g_k^\ell)_{k\in\mathbb{Z}}$ and $(\gamma_k^{\ell,\ell,\ell})_{k\in\mathbb{Z}}$ (see §6.4). But since $\gamma_k^{\ell,\ell,\ell}$ is non-zero only for $k \in \{-1, 0\}$ (cf. Remark 4.3), $\Gamma_{\ell,\ell}$ is explicitly available by

$$\Gamma_i^{\ell,\ell} = \frac{\sqrt{h_\ell}}{2}\left(g_i^\ell + g_{i-1}^\ell\right) \qquad \text{for all } i \in \mathbb{Z}. \tag{5.2}$$

Step 1b Next we compute $\Gamma_{\ell',\ell} = (\Gamma_i^{\ell',\ell})_{i\in\mathbb{Z}}$ for $\ell' = \ell-1, \ell-2, \ldots, 0$.

Lemma 5.1 *For $0 \le \ell' < \ell$, the relation*

$$\Gamma_i^{\ell',\ell} = \Gamma_{2i}^{\ell'+1,\ell} + \frac{1}{2}\left(\Gamma_{2i-1}^{\ell'+1,\ell} + \Gamma_{2i+1}^{\ell'+1,\ell}\right) \qquad \textit{for all } i \in \mathbb{Z} \tag{5.3}$$

can be used for its computation.

Proof. We use the recursions (4.10a,b) for $G_{i,j}^{\ell'',\ell',\ell}$:

$$\begin{aligned}
\Gamma_i^{\ell',\ell} &\underset{(4.11)}{=} G_{i,0}^{\ell',\ell',\ell} \underset{(4.10a)}{=} \frac{1}{\sqrt{2}}\left(G_{2i,0}^{\ell'+1,\ell',\ell} + G_{2i+1,0}^{\ell'+1,\ell',\ell}\right)\\
&\underset{(4.10b)}{=} \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}\left(G_{2i,0}^{\ell'+1,\ell'+1,\ell} + G_{2i,1}^{\ell'+1,\ell'+1,\ell}\right) + \frac{1}{\sqrt{2}}\left(G_{2i+1,0}^{\ell'+1,\ell'+1,\ell} + G_{2i+1,1}^{\ell'+1,\ell'+1,\ell}\right)\right)\\
&\underset{(4.9)}{=} \frac{1}{2}\left(G_{2i,0}^{\ell'+1,\ell'+1,\ell} + G_{2i-1,0}^{\ell'+1,\ell'+1,\ell} + G_{2i+1,0}^{\ell'+1,\ell'+1,\ell} + G_{2i,0}^{\ell'+1,\ell'+1,\ell}\right)\\
&\underset{(4.11)}{=} \Gamma_{2i}^{\ell'+1,\ell} + \frac{1}{2}\left(\Gamma_{2i-1}^{\ell'+1,\ell} + \Gamma_{2i+1}^{\ell'+1,\ell}\right).
\end{aligned}$$

■

Hence, $\Gamma_i^{\ell',\ell} = \sum_{k\in\mathbb{Z}} g_k^\ell \gamma_{k-i2^{\ell-\ell'}}^{\ell',\ell',\ell}$ $(\ell' < \ell)$ can be computed without referring to the data g_k^ℓ.

5.1.2 Step 2a

Let ℓ' be any level in the interval $[0, \ell]$, i.e., $\ell' \le \ell$. Then for each $\ell'' = \ell', \ell'-1, \ldots, 0$ the projection $P_{\ell''}(f_{\ell'} * g_\ell)$ is to be computed (see Problem 3.1). Following (4.2) and (4.3), the coefficients $\omega_i^{\ell''} = \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^\ell \gamma_{i,j,k}^{\ell'',\ell',\ell}$ are needed. The sequence is denoted by $\omega_{\ell''} = (\omega_i^{\ell''})_{i\in\mathbb{Z}}$.

For the starting value $\ell'' = \ell'$ we have

$$\omega_i^{\ell'} = \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^\ell \gamma_{i,j,k}^{\ell',\ell',\ell} \underset{(4.6)}{=} \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^\ell \gamma_{k-(i-j)2^{\ell-\ell'}}^{\ell',\ell',\ell} \underset{(4.11)}{=} \sum_{j\in\mathbb{Z}} f_j^{\ell'} \Gamma_{i-j}^{\ell',\ell} \qquad \text{for all } i \in \mathbb{Z}.$$

The sum $\sum_{j\in\mathbb{Z}} f_j^{\ell'} \Gamma_{i-j}^{\ell',\ell}$ describes the discrete convolution of the sequences[5] $f_{\ell'} := (f_j^{\ell'})_{j\in\mathbb{Z}}$ and $\Gamma_{\ell',\ell} := (\Gamma_k^{\ell',\ell})_{k\in\mathbb{Z}}$. Concerning the performance of the discrete convolution

$$\omega_{\ell'} = f_{\ell'} * \Gamma_{\ell',\ell} \tag{5.4}$$

we refer to Section 6.

5.1.3 Step 2b

Given $\omega_{\ell'}$ from (5.4), we compute $\omega_{\ell''}$ for $\ell'' = \ell'-1,\ldots,0$ by the following recursion formula.

Lemma 5.2 *The recursion*

$$\omega_i^{\ell''} = \frac{1}{\sqrt{2}}\left(\omega_{2i}^{\ell''+1} + \omega_{2i+1}^{\ell''+1}\right) \qquad \text{for all } i \in \mathbb{Z} \tag{5.5}$$

holds for all $0 \le \ell'' \le \ell'$.

Proof. Note that

$$\begin{aligned}\omega_i^{\ell''} &\underset{(4.3)}{=} \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \gamma_{i,j,k}^{\ell'',\ell',\ell} \underset{(4.4a)}{=} \frac{1}{\sqrt{2}} \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \left(\gamma_{2i,j,k}^{\ell''+1,\ell',\ell} + \gamma_{2i+1,j,k}^{\ell''+1,\ell',\ell}\right) \\ &= \frac{1}{\sqrt{2}}\left(\sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \gamma_{2i,j,k}^{\ell''+1,\ell',\ell} + \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \gamma_{2i+1,j,k}^{\ell''+1,\ell',\ell}\right) \underset{(4.3)}{=} \frac{1}{\sqrt{2}}\left(\omega_{2i}^{\ell''+1} + \omega_{2i+1}^{\ell''+1}\right).\end{aligned}$$

■

5.1.4 Intertwining the Computations for all $\ell'' \le \ell' \le \ell$

The superindex ℓ in $\Gamma_i^{\ell',\ell}$ indicates that this sequence at level ℓ' is originating from the data g_ℓ. Since the further treatment of $\Gamma_i^{\ell',\ell}$ does not depend on ℓ, we can gather all $\Gamma_i^{\ell',\ell}$ into

$$\Gamma_i^{\ell'} := \sum_{\ell=\ell'}^{L^g} \Gamma_i^{\ell',\ell} \qquad (0 \le \ell' \le L^g)\,. \tag{5.6}$$

Hence, their computation is performed by the loop[6]

for $\ell' := L^g$ **downto** 0 **do**	*explanations:*	
begin if $\ell' = L^g$ **then** $\Gamma_i^{L^g} := 0$	starting value,	
else $\Gamma_i^{\ell'} := \Gamma_{2i}^{\ell'+1} + \frac{1}{2}\left(\Gamma_{2i-1}^{\ell'+1} + \Gamma_{2i+1}^{\ell'+1}\right)$;	recursion (5.3),	(5.7)
$\Gamma_i^{\ell'} := \Gamma_i^{\ell'} + \Gamma_i^{\ell',\ell'}$	$\Gamma_i^{\ell',\ell'}$ defined in (5.2),	
end;	all $\Gamma_i^0, \Gamma_i^1, \ldots, \Gamma_i^{L^g}$ defined.	

[5] We use the same symbol $f_{\ell'}$ for the sequence $f_{\ell'} := (f_j^{\ell'})_{j\in\mathbb{Z}}$ and for the function $\sum_{j\in\mathbb{Z}} f_j^{\ell'} \Phi_j^{\ell'}$ from $\mathcal{M}_\ell$. The reference to the "discrete convolution" shows that here the interpretation as sequence is required.

[6] In this notation, $\Gamma_i^{\ell'}$ stands for the whole sequence $\Gamma_{\ell'} = (\Gamma_i^{\ell'})_{i\in\mathbb{Z}}$. More precisely, the computations have to be performed for i belonging to the support of $\Gamma_{\ell'}$.

Having available $\Gamma_i^{\ell'}$ for all $0 \le \ell' \le L^g$, we can compute $\omega_{\ell'}$ for any ℓ' (cf. (5.4)). For a moment, we use the symbols $\omega_{\ell',\ell'}, \omega_{\ell'-1,\ell'}, \ldots, \omega_{\ell'',\ell'}$ for the quantities computed in Step 2a,b. Here, the additional second index ℓ' expresses the fact that the data stem from $f_{\ell'}$ (see (5.4)).

The coarsening $\omega_{\ell',\ell'} \mapsto \omega_{\ell'-1,\ell'} \mapsto \ldots \mapsto \omega_{\ell'',\ell'}$ can again be done jointly for the different ℓ', i.e., we form

$$\omega_{\ell''} := \sum_{\ell'=\ell''}^{L^f} \omega_{\ell'',\ell'} \qquad \left(0 \le \ell'' \le \min\{L^\omega, L^f, L^g\}\right).$$

The algorithmic form is

	explanations:	
for $\ell'' := \min\{L^f, L^g\}$ **downto** 0 **do**		
begin if $\ell'' = \min\{L^f, L^g\}$ **then** $\omega_{\ell''} := 0$	starting value,	
else $\omega_i^{\ell''} := \frac{1}{\sqrt{2}}\left(\omega_{2i}^{\ell''+1} + \omega_{2i+1}^{\ell''+1}\right)$;	recursion (5.5),	(5.8)
$\omega_{\ell''} := \omega_{\ell''} + \omega_{\ell'',\ell''}$	$\omega_{\ell'',\ell''} = f_{\ell''} * \Gamma_{\ell''}$ defined in (5.4),	
end;	all $\omega_0, \omega_1, \ldots, \omega_{\min\{L^f,L^g\}}$ defined.	

Note that this algorithm yields[7]

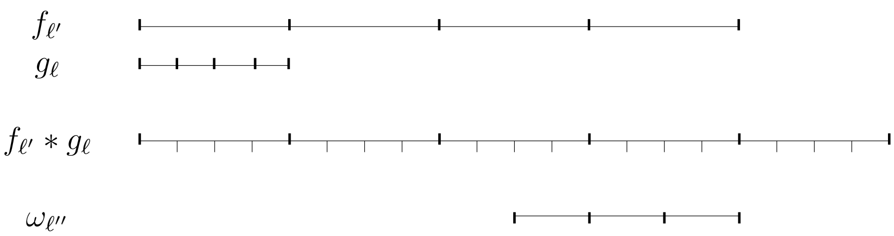
$$\omega_{\ell''} = P_{\ell''}\left(\sum_{\ell=\ell''}^{L^g} \sum_{\ell'=\ell''}^{\ell} f_{\ell'} * g_\ell\right) \qquad \left(0 \le \ell'' \le \min\{L^\omega, L^f, L^g\}\right)$$

involving all combinations of indices with $\ell'' \le \ell' \le \ell$.

Note that, in addition, the contribution $\omega_{\ell''} = P_{\ell''}\left(\sum_{\ell=\ell''+1}^{L^g} \sum_{\ell'=\ell''+1}^{\ell} g_{\ell'} * f_\ell\right)$ corresponding to the second sum in (3.5) has to be computed by the same procedure with g and f interchanged.

5.2 Case B: $\ell' < \ell'' \le \ell$

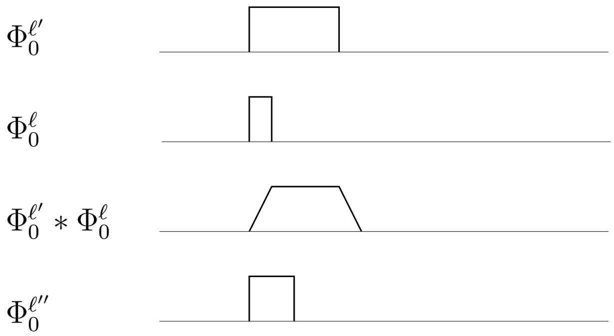
Case B is illustrated by the following figure.



Differently from Case A, the step size $h_{\ell''}$ used by the projection $P_{\ell''}$ is smaller than the step size $h_{\ell'}$ but larger than h_ℓ.

The basis functions related to Case B are shown below[4].

[7]The restriction $\ell'' \le \min\{L^\omega, L^f, L^g\}$ follows from $\ell'' \le \ell' \le \ell$, since $\ell' \le L^f$, $\ell \le L^g$, and $\ell'' \le L^\omega$.



5.2.1 Explanations for $\ell'' = \ell' + 1$

We will use a loop of ℓ'' from $\ell' + 1$ to ℓ. Here we discuss the first value $\ell'' = \ell' + 1$ and assume $\ell' + 1 \le \ell$.

The function $f_{\ell'} = \sum_j f_j^{\ell'} \Phi_j^{\ell'}$ can be transformed into a function of level $\ell' + 1$ by using (2.8):

$$f_{\ell'} = \sum_j \hat{f}_j^{\ell'+1} \Phi_j^{\ell'+1} \qquad \text{with } \hat{f}_{2j}^{\ell'+1} := \hat{f}_{2j+1}^{\ell'+1} := \frac{1}{\sqrt{2}} f_j^{\ell'}. \tag{5.9}$$

Let $\hat{f}_{\ell'+1} := \left(\hat{f}_j^{\ell'+1}\right)_{j\in\mathbb{Z}}$ be the sequence of the newly defined coefficients. Since $\ell'' = \ell'+1 \le \ell$, the three level numbers $\ell'', \ell'+1, \ell$ satisfy the inequalities of Case A. As in Step 2a of Case A (see §5.1.2) the desired coefficients of the projection at level $\ell'' = \ell' + 1$ are $\omega_i^{\ell'+1} = \sum_{j\in\mathbb{Z}} \hat{f}_j^{\ell'+1} \Gamma_{i-j}^{\ell'+1,\ell}$, i.e., the discrete convolution $\omega_{\ell'+1} = \hat{f}_{\ell'+1} * \Gamma_{\ell'+1,\ell}$ is to be performed.

In the given formulation, the reinterpretation of $f_{\ell'}$ as a function of level $\ell' + 1$ seems dangerous, since by (5.9) the number of coefficients is doubled. If we repeat this procedure up to level L, the number of coefficients would be multiplied by $2^{L-\ell'}$. The remedy is a restriction of (5.9) to those coefficients $\hat{f}_j^{\ell'+1}$ which are really needed. The coefficients $\omega_i^{\ell'+1}$ are required only for $i \in \mathcal{I}_\ell^\omega$ (cf. (3.1)), say for $i \in \{i_1^\omega, \ldots, i_2^\omega\}$. Let the nonzero coefficients $\Gamma_j^{\ell'+1,\ell}$ lie in $i_1^\Gamma \le j \le i_2^\Gamma$. The sum in $\omega_i^{\ell'+1} = \sum_{j\in\mathbb{Z}} \hat{f}_j^{\ell'+1} \Gamma_{i-j}^{\ell'+1,\ell}$ for $i \in \{i_1^\omega, \ldots, i_2^\omega\}$ involves only $\hat{f}_j^{\ell'+1}$-coefficients with $i_1^\omega - i_2^\Gamma \le j \le i_2^\omega - i_1^\Gamma$. Hence, the number of $\hat{f}_j^{\ell'+1}$-coefficients is bounded by $i_2^\omega - i_1^\omega + i_2^\Gamma - i_1^\Gamma + 1$. A similar number appears for later levels.

Since in the further recursions also $\omega_i^{\ell''}$ for $\ell'' = \ell' + 2, \ldots, \min\{L^\omega, L^g\}$ are to be determined, the interval $[i_1^\omega, i_2^\omega]$ from above is to be increased a bit (using the notation of §6.1, we have to replace $S_c(\omega_{\ell'+1})$ by $S_{cc}(\omega_{\ell'+1})$).

5.2.2 Complete Recursion

Steps 1a,b in Case A have already produced the coefficients $\Gamma_j^{\ell'}$ gathering all $\Gamma_j^{\ell',\ell}$ ($\ell \ge \ell'$, cf. (5.6)). For $\ell'' = \ell' + 1, \ell' + 2, \ldots, \ell$ we represent the function $f_{\ell'}$ at these levels ℓ'' by computing the coefficients $\hat{f}_j^{\ell''}$ as in (5.9):

$$\hat{f}_j^{\ell'} := f_j^{\ell'} \qquad \text{(starting value)}, \tag{5.10a}$$

$$\hat{f}_{2j}^{\ell''} := \hat{f}_{2j+1}^{\ell''} := \frac{1}{\sqrt{2}} \hat{f}_j^{\ell''-1} \qquad (\ell' + 1 \le \ell'' \le \ell)\,. \tag{5.10b}$$

Note, however, that only those coefficients are to be determined which are really needed in the next step, which is the discrete convolution

$$\omega_{\ell''} = \hat{f}_{\ell''} * \Gamma_{\ell''} \qquad (\ell' + 1 \le \ell'' \le \ell) \tag{5.10c}$$

of the sequence $\hat{f}_{\ell''} := (\hat{f}_j^{\ell''})_{j\in\mathbb{Z}}$ with $\Gamma_{\ell''}$.

5.2.3 Combined Computations for all $\ell' < \ell'' \le \ell$

The algorithm is

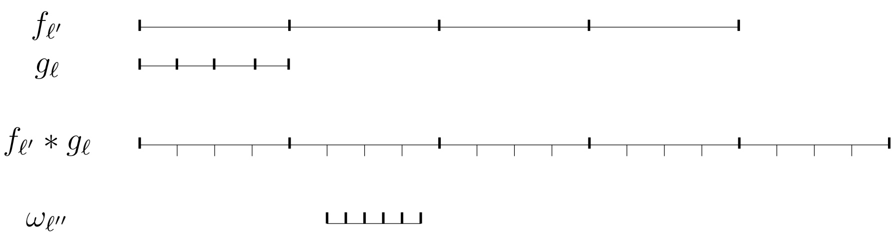
	explanations:	
$\hat{f}_j^0 := 0;$		
for $\ell'' := 1$ **to** $\min\{L^\omega, L^g\}$ **do**		
begin $\hat{f}_j^{\ell''-1} := \hat{f}_j^{\ell''-1} + f_j^{\ell''-1};$	starting value (5.10a),	(5.11)
$\quad\hat{f}_{2j}^{\ell''} := \hat{f}_{2j+1}^{\ell''} := \hat{f}_j^{\ell''-1}/\sqrt{2};$	see (5.10b),	
$\quad\omega_{\ell''} := \hat{f}_{\ell''} * \Gamma_{\ell''}$	see (5.10c)	
end;		

The limitation by $\ell'' \le \min\{L^\omega, L^g\}$ in line 2 is correct, since for $\ell'' > L^\omega$ no $\omega_{\ell''}$ are required, while for $\ell'' > L^g$ the sequence $\Gamma_{\ell''}$ is not defined (i.e., formally $\Gamma_{\ell''} = 0$).

The sum $\hat{f}_j^{\ell''-1} + f_j^{\ell''-1}$ in the third line defines $\hat{f}_j^{\ell''-1}$ as coefficients of $\sum_{\ell'=0}^{\ell''-1} f_{\ell'} = \sum_j \hat{f}_j^{\ell''-1}\Phi_j^{\ell''-1}$. Therefore the next two lines consider all combinations of $\ell' < \ell''$. Since $\Gamma_{\ell''}$ contains all contributions from $\ell \ge \ell''$, $\omega_{\ell''}$ is the projection $P_{\ell''}(\sum_{\ell',\ell \text{ with } \ell'<\ell''\le\ell} f_{\ell'} * g_\ell)$.

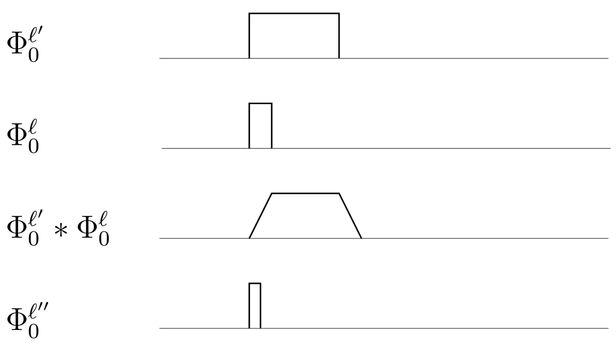
5.3 Case C: $\ell' \le \ell < \ell''$

The following figure illustrates Case C.



Now the step size $h_{\ell''}$ used by the projection $P_{\ell''}$ is smaller than both $h_{\ell'}$ and h_ℓ.

The basis functions related to Case C are sketched below[4].



5.3.1 Explanations

This case is completely different from the previous cases. The exact convolution

$$\omega_{\mathrm{exact}}(x) := \int f_{\ell'}(y) g_\ell(x-y)\mathrm{d}y \qquad (x \in \mathbb{R})$$

is a piecewise linear and globally continuous function with possible jumps of the derivative at the grid points νh_ℓ ($\nu \in \mathbb{Z}$) of the grid at level ℓ. The projection $P_{\ell''}\omega_{\mathrm{exact}} = \sum_i \omega_i^{\ell''} \Phi_i^{\ell''}$ requires all scalar products

$$\omega_i^{\ell''} = \int \Phi_i^{\ell''}(x)\omega_{\mathrm{exact}}(x)\mathrm{d}x\,.$$

Note that the whole support of $\Phi_i^{\ell''}$ belongs to one of the intervals $[\nu h_\ell, (\nu+1)h_\ell]$, where $\omega_{\mathrm{exact}}(x)$ is an affine linear function. Consider, e.g., the interval $[0, h_\ell]$ and note that $\mathrm{supp}(\Phi_i^{\ell''}) \subset [0, h_\ell]$ holds if and only if $0 \le i \le 2^{\ell''-\ell} - 1$. Hence, $2^{\ell''-\ell}$ functionals of $\omega_{\mathrm{exact}}|_{[0,h_\ell]}$ are to be evaluated. Since the space of affine linear functions has dimension 2, there are not more than 2 linearly independent functionals! It suffices to compute the two functionals

$$\alpha_0 := \int \Phi_0^{\ell''}(x)\omega_{\mathrm{exact}}(x)\mathrm{d}x, \qquad \beta_0 := \int \Phi_{2^{\ell''-\ell}-1}^{\ell''}(x)\omega_{\mathrm{exact}}(x)\mathrm{d}x.$$

Because of $\alpha_0 = \sqrt{h_{\ell''}}\,\omega_{\mathrm{exact}}(h_{\ell''}/2)$ and $\beta_0 = \sqrt{h_{\ell''}}\,\omega_{\mathrm{exact}}(h_\ell - h_{\ell''}/2)$, we obtain all functionals by linear interpolation:

$$\omega_i^{\ell''} = \alpha_0 + \frac{i}{2^{\ell''-\ell}-1}(\beta_0 - \alpha_0) \qquad \text{for } 0 \le i \le 2^{\ell''-\ell} - 1.$$

Similarly, values α_ν and β_ν have to be computed for each interval $[\nu h_\ell, (\nu+1)h_\ell]$, where the evaluation of $P_{\ell''}\omega_{\mathrm{exact}}$ is desired. The computation of the sequences $\alpha = (\alpha_i)_{i\in\mathbb{Z}}$ and $\beta = (\beta_i)_{i\in\mathbb{Z}}$ corresponds to the previous case $\ell' \le \ell'' = \ell$. The only difference is that in the definition of $\gamma_{i,j,k}^{\ell,\ell',\ell} = \iint \Phi_i^\ell(x)\Phi_j^{\ell'}(y)\Phi_k^\ell(x-y)\mathrm{d}x\mathrm{d}y$ one has to replace Φ_i^ℓ by $\Phi_{i2^{\ell''-\ell}}^{\ell''}$ for the computation of α_i and by $\Phi_{2^{\ell''-\ell}(i+1)-1}^{\ell''}$ for the computation of β_i.

We do not give further details, since we recommend another approach which reduces the cost by one half.

5.3.2 Pointwise Evaluations

Since ω_{exact} is continuous in $\mathbb{R}$, the point values

$$\delta_\nu^\ell := \omega_{\mathrm{exact}}(\nu h_\ell) \qquad \text{for } \nu \in \mathbb{Z}$$

are well-defined. For each interval $[\nu h_\ell, (\nu+1)h_\ell]$ we have two functionals δ_ν^ℓ and $\delta_{\nu+1}^\ell$, but since each δ_ν^ℓ serves for two adjacent intervals, this approach is cheaper.

The computation of the sequence $\delta^\ell = \left(\delta_\nu^\ell\right)_{\nu\in\mathbb{Z}}$ will be described in §5.3.3. Here we assume that we have the values δ_ν^ℓ at the points of interest. Then we get the desired $\omega_i^{\ell''}$ as follows. Write the index i as

$$i = \nu 2^{\ell''-\ell} + \mu \qquad \text{with } 0 \le \mu \le 2^{\ell''-\ell} - 1.$$

Then

$$\omega_i^{\ell''} = \int \Phi_i^{\ell''}(x)\omega_{\text{exact}}(x)\mathrm{d}x = \frac{1}{\sqrt{h_{\ell''}}}\int_{ih_{\ell''}}^{(i+1)h_{\ell''}} \omega_{\text{exact}}(x)\mathrm{d}x = h_{\ell''}\frac{\omega_{\text{exact}}\left((i+\frac{1}{2})h_{\ell''}\right)}{\sqrt{h_{\ell''}}}$$
$$= \sqrt{h_{\ell''}}\,\omega_{\text{exact}}\left(\nu h_\ell + (\mu+\tfrac{1}{2})h_{\ell''}\right) = \sqrt{h_{\ell''}}\left(\delta_\nu^\ell + \frac{\mu+\frac{1}{2}}{2^{\ell''-\ell}}\left(\delta_{\nu+1}^\ell - \delta_\nu^\ell\right)\right). \tag{5.12}$$

Obviously, this formula holds for all levels $\ell'' > \ell$ simultaneously. In the particular case of $\ell'' = \ell+1$, the previous formula becomes

$$\omega_i^{\ell''} = \sqrt{h_{\ell+1}}\left(\delta_\nu^\ell + \frac{\mu+\frac{1}{2}}{2}\left(\delta_{\nu+1}^\ell - \delta_\nu^\ell\right)\right) \qquad \text{for } i = 2\nu+\mu,\ \mu\in\{0,1\}. \tag{5.13}$$

5.3.3 Computation of δ

We define new $\check{\gamma}$-coefficients

$$\check{\gamma}_{i,j,k}^\ell := \int \Phi_j^\ell(y)\Phi_k^\ell(ih_\ell - y)\mathrm{d}y \qquad (i,j,k\in\mathbb{Z})$$

($\Phi_i^\ell(x)$ from (4.1) replaced by the Dirac function at ih_ℓ), which involves only one level ℓ. Simple substitutions yield

$$\begin{aligned}\check{\gamma}_{i,j,k}^\ell &= \int \Phi_0^\ell(y-jh_\ell)\Phi_0^\ell((i-k)\,h_\ell - y)\mathrm{d}y \\ &= \int \Phi_0^\ell(y)\Phi_0^\ell((i-k)\,h_\ell - (y+jh_\ell))\mathrm{d}y = \int \Phi_0^\ell(y)\Phi_0^\ell\left((i-k-j)h_\ell - y\right)\mathrm{d}y \\ &= \int \Phi_0^\ell(y)\Phi_{k-i+j2^{\ell-\ell'}}^\ell(-y)\mathrm{d}y = \check{\gamma}_{k-i+j}^\ell\end{aligned}$$

for the "simplified" coefficient $\check{\gamma}_\nu^\ell := \check{\gamma}_{0,0,\nu}^\ell$.

The δ-values of $f_\ell * g_\ell$ are

$$\begin{aligned}\delta_i^\ell &= (f_\ell * g_\ell)\,(ih_\ell) = \sum_{j,k\in\mathbb{Z}} f_j^\ell\, g_k^\ell \int \Phi_j^\ell(y)\Phi_k^\ell(ih_\ell - y)\mathrm{d}y = \sum_{j,k\in\mathbb{Z}} f_j^\ell\, g_k^\ell\, \check{\gamma}_{i,j,k}^\ell \\ &= \sum_{j\in\mathbb{Z}} f_j^\ell\, \check{G}_{i,j}^\ell, \qquad \text{where } \check{G}_{i,j}^\ell := \sum_{k\in\mathbb{Z}} g_k^\ell\, \check{\gamma}_{i,j,k}^\ell.\end{aligned}$$

$\check{\gamma}_{i,j,k}^\ell = \check{\gamma}_{k-i+j}^\ell$ shows $\check{G}_{i,j}^\ell = \check{G}_{i-j,0}^\ell$, so that $\delta_i^\ell = \sum_{j\in\mathbb{Z}} f_j^\ell\, \check{G}_{i-j,0}^\ell$ becomes a discrete convolution. Since the $\check{\gamma}$-values are rather simple:

$$\check{\gamma}_k^\ell = \begin{cases} 1 & \text{for } k = -1, \\ 0 & \text{otherwise,}\end{cases}$$

$\check{G}_{i-j,0}^\ell = g_{i-j-1}^\ell$ holds, leading to the direct representation

$$\delta_i^\ell = \sum_{j\in\mathbb{Z}} f_j^\ell\, g_{i-j-1}^\ell. \tag{5.14}$$

The values δ_i^ℓ belong to level ℓ in the sense that $\delta_i^\ell = (f_\ell * g_\ell)\,(ih_\ell)$ is evaluated in the mesh of size h_ℓ. We can evaluate the same function at levels $\ell'' > \ell$:

$$\delta_i^{\ell''} = (f_\ell * g_\ell)\,(ih_{\ell''}).$$

Since $f_\ell * g_\ell$ is linear in $I_i^\ell = [ih_\ell, (i+1)h_\ell)$, the obvious recursion is

$$\delta_{2i}^{\ell''} = \delta_i^{\ell''-1}, \qquad \delta_{2i+1}^{\ell''} = \frac{1}{2}\left(\delta_i^{\ell''-1} + \delta_{i+1}^{\ell''-1}\right) \qquad \text{for } \ell'' > \ell. \tag{5.15}$$

5.3.4 Combined Computations for all $\ell' \le \ell < \ell''$

for $\ell := 0$ **to** $L^\omega - 1$ **do**
begin if $\ell = 0$ **then begin** $\hat{f}_i^0 := 0$; $\hat{\delta}_i^0 := 0$ **end else**
  begin compute $\hat{f}_i^\ell$ from $\hat{f}_i^{\ell-1}$ by (5.10b);
    compute $\hat{\delta}_i^\ell$ from $\hat{\delta}_i^{\ell-1}$ by (5.15)
  end;
  if $\ell \le L^f$ **then** $\hat{f}_i^\ell := \hat{f}_i^\ell + f_i^\ell$;
  if $\ell \le L^g$ **then**   (5.16)
  begin compute δ_i^ℓ by the convolution $\sum_{j\in\mathbb{Z}} \hat{f}_j^\ell\, g_{i-j-1}^\ell$;
    $\hat{\delta}_i^\ell := \hat{\delta}_i^\ell + \delta_i^\ell$;
  end;
  compute $\omega_i^{\ell+1}$ from $\hat{\delta}_i^\ell$ by (5.13)
end;

The quantities $\hat{f}_i^\ell$ obtained in line 6 of the algorithm are the coefficients of $\sum_{\ell'=0}^{\ell} f_{\ell'} = \sum_i \hat{f}_i^\ell \Phi_i^\ell$. Therefore the δ_i^ℓ from line 8 are the evaluations of $\sum_{\ell'=0}^{\ell} f_{\ell'} * g_\ell$ at ih_ℓ. The quantities $\hat{\delta}_i^\ell$ updated in line 9 contain all levels from 0 to the actual ℓ. Hence, the $\omega_i^{\ell+1}$ in line 11 belong to the projection $P_{\ell''}\left(\sum_{\ell',\lambda \text{ with } 0\le\ell'\le\lambda\le\ell} f_{\ell'} * g_\lambda\right)$ at level $\ell'' = \ell + 1$, where ℓ is the actual value of the loop index.

5.4 Range of Products

In the previous subsections we have reduced the problem to a number of specific discrete convolutions (the first example is (5.4)). The resulting products are infinite sequences $(c_\nu)_{\nu\in\mathbb{Z}}$. The first reasonable reduction would be to determine $(c_\nu)_{\nu=\nu_1}^{\nu_2}$ only in the support $[\nu_1, \nu_2] \cap \mathbb{Z}$ of the sequence. But it is essential to go a step further. Even if we need the function $f_{\ell'} * g_\ell$ (see (3.5)) in the whole support $S := \operatorname{supp}(f_{\ell'} * g_\ell)$, the projections $P_{\ell''}(f_{\ell'} * g_\ell)$ are required in disjoint subsets $S_{\ell''} \subset S$. In terms of the sequences $(c_\nu)_{\nu\in\mathbb{Z}}$ this means that we are interested in the components c_ν in an index interval $[\nu_1', \nu_2'] \cap \mathbb{Z}$ which is possibly much smaller than the support $[\nu_1, \nu_2]$.

The analysis of the cost in §7 will assume that *only the necessary parts* are evaluated. Therefore the restriction of the evaluation to the minimal range of the discrete product $(c_\nu)_{\nu\in\mathbb{Z}}$ is an essential part of the algorithm. The treatment of this information in the fast discrete convolution will be explained in §6.6.

6 Discrete Convolution and Applications

The basic tool for the discrete convolution $c := a * b$ is the well-known FFT. Here, we check the cost in dependence of the size of the factors a, b and the required part of the product c. In the following subsection, we first introduce measures of the data size.

6.1 Data Sizes

6.1.1 Definitions of $S_d(f_\ell)$, $N_d(f_\ell)$ and $S_c(f_\ell)$, $N_c(f_\ell)$

We use a notation like f_ℓ ambiguously for the function $\sum_{i\in\mathbb{Z}} f_i^\ell \Phi_i^\ell$ as well as for the (infinite) sequence $f_\ell = \left(f_i^\ell\right)_{i\in\mathbb{Z}}$. In both cases the *discrete support* of f_ℓ is

$$S_d(f_\ell) := \left\{i \in \mathbb{Z} : f_i^\ell \neq 0\right\}$$

with the corresponding cardinality

$$N_d(f_\ell) := \#S_d(f_\ell)$$

(the subindex "d" indicates "discrete support").

Unfortunately, the computational cost often depends on the *convex hull* of $S_d(f_\ell)$, which is

$$S_c(f_\ell) = \{i_{\min}, i_{\min}+1, \ldots, i_{\max}\} \quad \text{with } i_{\min} := \min_{i\in S_d(f_\ell)} i,\ i_{\max} := \max_{i\in S_d(f_\ell)} i, \tag{6.1}$$

and on the number

$$N_c(f_\ell) := \#S_c(f_\ell) = i_{\max} - i_{\min} + 1$$

(here, the subindex "c" indicates "convex hull").

The definitions from above require a non-empty set $S_d(f_\ell)$. Formally, we set $S_c(f_\ell) = \emptyset$ and $N_c(f_\ell) = 0$ for the uninteresting case of $S_d(f_\ell) = \emptyset$.

In the case of discrete convolutions, we often use the symbols a, b, c for the sequences: $c = a*b$. Correspondingly, the supports and sizes are $S_c(a) = \{i \in \mathbb{Z} : a_i \neq 0\}$, $S_d(a)$, $N_c(a)$, $N_d(a)$, etc.

6.1.2 Definitions of $S_{dd}(f_\ell)$, $N_{dd}(f_\ell)$ and $S_{cc}(f_\ell)$, $N_{cc}(f_\ell)$

Next, we define a modification of S_d, S_c and N_c, N_d, which we denote by doubled subscripts:

$$S_{dd}(f_\ell) := \left\{i \in \mathbb{Z} : I_{i_{\ell'}}^{\ell'} \subset I_i^\ell \text{ for some } \ell' \geq \ell \text{ and } i_{\ell'} \in S_d(f_{\ell'})\right\}.$$

The intervals I_i^ℓ with[8] $i \in S_{dd}(f_\ell)$ are those which are either used at level ℓ by f_ℓ or refined into smaller intervals used by $f_{\ell'}$, $\ell' \geq \ell$. Therefore, an equivalent definition is that $S_{dd}(f_\ell)$ is the smallest set so that $\bigcup_{i\in S_{dd}(f_\ell)} I_i^\ell$ contains the support of the function $\sum_{\ell'=\ell}^{L} f_{\ell'}$. Another explicit definition is

$$S_{dd}(f_\ell) = \bigcup_{\ell'=\ell}^{L} \left\{\left\lfloor i2^{\ell-\ell'}\right\rfloor : i \in S_d(f_{\ell'})\right\}. \tag{6.2}$$

[8]In fact, the notation $S_{dd}(f_\ell, f_{\ell+1}, \ldots, f_L)$ would be more precise, since this set depends on all data $f_\ell, f_{\ell+1}, \ldots, f_L$. However, we avoid this long expression.

In the case of Figure (1.2b), $S_d(f_0)$ is contained in $\{4,5,6,7\}$ corresponding to $[1/2,1)$, while $S_{dd}(f_0)$ is contained in $\{0,1,\ldots,7\}$ corresponding to $[0,1)$.

As above N_{dd} denotes the cardinality of S_{dd}, while $S_{cc}(f_\ell)$ – as is (6.1) – is the convex hull of S_{dd} and N_{dd} is its cardinality:

$$N_{dd}(f_\ell) := \#S_{dd}(f_\ell), \quad S_{cc}(f_\ell) := \left[\min_{i\in S_{dd}(f_\ell)} i, \max_{i\in S_{dd}(f_\ell)} i\right] \cap \mathbb{Z}, \quad N_{cc}(f_\ell) := \#S_{cc}(f_\ell).$$

The characterisation of $S_{dd}(f_\ell)$ by (6.2) shows that roughly

$$N_{dd}(f_\ell) \lesssim \sum_{\ell'=\ell}^{L} 2^{\ell-\ell'} N_d(f_{\ell'}). \tag{6.3}$$

6.1.3 Meaning of $S_d(\omega_\ell)$ etc.

In order not to introduce other notations, we use the notations $S_d,\ldots,N_{cc}$ also for the argument ω_ℓ with the following meaning:

$$S_d(\omega_{\ell''}) := \mathcal{I}^{\omega}_{\ell''}. \tag{6.4}$$

The function or sequence $\omega_{\ell''}$ is obtained, e.g., by the projected convolution $P_{\ell''}(f_{\ell'} * g_\ell)$. Although the support may be larger, we only consider the coefficients $\omega_i^{\ell''}$ for $i \in \mathcal{I}^{\omega}_{\ell''}$. If we redefine $\omega_i^{\ell''} := 0$ for $i \notin \mathcal{I}^{\omega}_{\ell''}$, the definition (6.4) coincides with the usual definition. The further notations $N_d(\omega_{\ell''})$, $S_c(\omega_{\ell''})$, $N_c(\omega_{\ell''})$, $S_{dd}(\omega_{\ell''})$, $N_{dd}(\omega_{\ell''})$, $S_{cc}(\omega_{\ell''})$, $N_{cc}(\omega_{\ell''})$ follow as before from $S_d(\omega_{\ell''})$.

6.1.4 Meaning of $N_d(f)$ etc.

The functions $f \in \mathcal{S}^f$ consists of the level contributions f_ℓ whose size is measured either by $N_d(f_\ell)$ or $N_c(f_\ell)$. The size of f is defined by their sums:

$$N_d(f) := \sum_{\ell=0}^{L} N_d(f_\ell), \;\; N_c(f) := \sum_{\ell=0}^{L} N_c(f_\ell), \;\; N_{dd}(f) := \sum_{\ell=0}^{L} N_{dd}(f_\ell), \;\; N_{cc}(f) := \sum_{\ell=0}^{L} N_{cc}(f_\ell).$$

In standard applications the supports of f_ℓ and $f_{\ell-1}$ are neighboured, so that there is no gap between $S_c(f_\ell)$ and $S_c(f_{\ell-1})$. In such a case, (6.3) can be extended to the convex supports:

$$N_{cc}(f_\ell) \lesssim \sum_{\ell'=\ell}^{L} 2^{\ell-\ell'} N_c(f_{\ell'}). \tag{6.5}$$

An immediate conclusion from this inequality is

$$N_{cc}(f) \lesssim \sum_{\ell=0}^{L}\sum_{\ell'=\ell}^{L} 2^{\ell-\ell'} N_c(f_{\ell'}) \le 2\sum_{\ell'=0}^{L} N_c(f_{\ell'}) = 2N_c(f).$$

6.2 Basic Convolution Problem

Let

$$a := (a_\nu)_{\nu\in\mathbb{Z}}\,, \qquad b := (b_\nu)_{\nu\in\mathbb{Z}}$$

be given sequences of real or complex numbers with support contained in $\{0, 1, \ldots, n-1\}$:

$$S_d(a) \subset \{0, 1, \ldots, n-1\}\,, \quad S_d(b) \subset \{0, 1, \ldots, n-1\}\,,$$

i.e., $a_\nu = b_\nu = 0$ for $\nu \notin \{0, 1, \ldots, n-1\}$. The third sequence $c := (c_\nu)_{\nu\in\mathbb{Z}}$ is defined as *discrete convolution* product:

$$c := a * b, \quad \text{i.e.,} \quad c_\nu := \sum_{\alpha\in\mathbb{Z}} a_\alpha b_{\nu-\alpha}\,.$$

Obviously,

$$S_d(c) \subset \{0, 1, \ldots, 2n-2\}\,.$$

Fix some $m \in \mathbb{Z}$ with $m \geq 2n-1$. By means of the *fast Fourier transform* one determines the coefficients $\hat{a}_\mu$, $\hat{b}_\mu$ of $\hat{a} = (\hat{a}_\mu)_{\mu=0}^{m-1}$ and $\hat{b} = \left(\hat{b}_\mu\right)_{\mu=0}^{m-1}$ such that

$$a_\nu = \sum_{\mu=0}^{m-1} \hat{a}_\mu e^{i\nu\mu 2\pi/m}, \quad b_\nu = \sum_{\mu=0}^{m-1} \hat{b}_\mu e^{i\nu\mu 2\pi/m} \qquad \text{for } 0 \leq \nu \leq m-1.$$

Since

$$\begin{aligned}
c_\nu &= \sum_{\alpha=0}^{m-1} a_\alpha b_{\nu-\alpha} = \sum_{\alpha=0}^{m-1}\sum_{\mu=0}^{m-1} \hat{a}_\mu\, e^{i\alpha\mu 2\pi/m} \sum_{\lambda=0}^{m-1} \hat{b}_\lambda e^{i(\nu-\alpha)\lambda 2\pi/m} \\
&= \sum_{\mu=0}^{m-1}\sum_{\lambda=0}^{m-1} \hat{a}_\mu \hat{b}_\lambda \sum_{\alpha=0}^{m-1} e^{i(\alpha\mu+(\nu-\alpha)\lambda)2\pi/m} = \sum_{\mu=0}^{m-1}\sum_{\lambda=0}^{m-1} \hat{a}_\mu \hat{b}_\lambda\, e^{i\nu\lambda 2\pi/m} \underbrace{\sum_{\alpha=0}^{m-1} e^{i\alpha(\mu-\lambda)2\pi/m}}_{m\delta_{\mu,\lambda}} \\
&= m \sum_{\mu=0}^{m-1} \hat{a}_\mu \hat{b}_\mu\, e^{i\nu\mu 2\pi/m} \qquad \text{for } \nu = 0, 1, \ldots, 2n-2,
\end{aligned}$$

m products $\hat{c}_\mu := m\hat{a}_\mu\hat{b}_\mu$ $(0 \leq \mu \leq m-1)$ and the back transformation $c_\nu = \sum_{\mu=0}^{m-1} \hat{c}_\mu\, e^{i\nu\mu\pi/m}$ $(0 \leq \nu \leq m-1)$ are to be computed.

Remark 6.1 *a) A reasonable choice of m is $m = 2^p$ with the smallest $p \in \mathbb{N}$ such that* $2^p \geq 2n-1$.
b) If $c_\nu = \sum_{\mu=0}^{m-1} \hat{c}_\mu e^{i\nu\mu\pi/m}$ is evaluated for $\nu \notin \{0, 1, \ldots, 2n-2\}$, this value is in general different from the true value $c_\nu = 0$.
c) The computational cost of the convolution $a, b \mapsto c = a * b$ is $\mathcal{O}(n \log n)$, provided that m is chosen $\leq Cn$ as, e.g., proposed in Part a).

6.3 General Finite Sequences

Let $a := (a_\nu)_{\nu\in\mathbb{Z}}$, $b := (b_\nu)_{\nu\in\mathbb{Z}}$ be general sequences with

$$n_a := N_c(a) < \infty, \quad n_b := N_c(b) < \infty. \tag{6.6}$$

Then, for suitable $i_a, i_b \in \mathbb{Z}$, we have $S_c(a) = \{i_a, i_a+1, \ldots, i_a+n_a-1\}$ and $S_c(b) = \{i_b, i_b+1, \ldots, i_b+n_b-1\}$. The support of $c := a*b$ has length $N_c(c) \le N_c(a) + N_c(b) - 1$ and is contained in

$$S_c(c) = \{i_c, i_c+1, \ldots, i_c+N_c(c)-1\} \qquad \text{with } i_c := i_a + i_b.$$

To return to the situation of §6.2, we introduce the shifted sequences

$$a', b' \qquad \text{with } a'_\nu := a_{\nu-i_a} \text{ and } b'_\nu := b_{\nu-i_b} \qquad (\nu \in \mathbb{Z})\,. \tag{6.7}$$

Together with

$$n := \max\{n_a, n_b\}$$

we can apply §6.2 and obtain $c' := a' * b'$. Then the desired convolution $c = a*b$ results from a back shift:

$$c_\mu = c'_{\mu+i_c} \qquad (\mu \in \mathbb{Z})\,.$$

According to $n := \max\{n_a, n_b\}$ and Remark 6.1c, the required work is

$$\mathcal{O}(\max\{n_a, n_b\}\log(\max\{n_a, n_b\})).$$

As we will see from §6.5, this complexity can be improved a little.

6.4 Conjugate Convolution

Sometimes, a sequence c with

$$c_\mu := \sum_{\nu\in\mathbb{Z}} a_\nu b_{\nu-\mu} \qquad (\mu \in \mathbb{Z})$$

appears. We may interpret this operation $c := a \bar{*} b$ as conjugate convolution, since

$$\langle a\bar{*}b, d\rangle = \langle a, b*d\rangle$$

holds ($\langle\cdot,\cdot\rangle$ is the Euclidean scalar product). Note that the operation $\bar{*}$ is not symmetric!

To return to the situation of §6.2, we introduce the sequence b' with $b'_\nu := b_{-\nu}$ and observe that $c = a * b'$.

6.5 Reduced Computational Cost for $N_c(b) < N_c(a)$

According to the two last subsections, we may assume without loss of generality the basic situation of §6.2.

The cost of the convolution $c = a * b$ given in Remark 6.1c is expressed in terms of n, which is at least $\max\{n_a, n_b\}$ with

$$n_a := N_c(a), \quad n_b := N_c(b). \tag{6.8a}$$

Next we consider the situation that n_a and n_b have quite different sizes. Since the convolution is symmetric, we may assume without loss of generality that $n_b \le n_a$. Note that we assume

$$S_c(a) = \{0, 1, \ldots, n_a - 1\} \qquad \text{and} \quad S_c(b) = \{0, 1, \ldots, n_b - 1\} \tag{6.8b}$$

as obtained after applying (6.7).

If n_b is a small constant, even the naive summation $\sum_{\alpha=0}^{n_b-1} b_\alpha a_{\nu-\alpha}$ may be performed for all $\nu \in S_c(c)$. Obviously, the overall cost is $\mathcal{O}(n_a)$ without any logarithmic factor.

$n_b \le n_a$ allows a factorisation $n_a = (q-1)\, n_b + r$ $(q, r \in \mathbb{N})$ with remainder $1 \le r \le n_b$. Then the data $a = (a_\nu)_{\nu=0}^{n_a-1}$ can be split into q packages

$$a^{(1)} = (a_\nu)_{\nu=0}^{n_b-1}, \; a^{(2)} = (a_\nu)_{\nu=n_b}^{2n_b-1}, \ldots, \; a^{(q-1)} = (a_\nu)_{\nu=(q-2)n_b}^{(q-1)n_b-1}, \; a^{(q)} = (a_\nu)_{\nu=(q-1)n_b}^{n_a-1},$$

where the components outside the given ranges are defined by zero. Note that $\sum_{\ell=0}^{q} a^{(\ell)} = a$ and that $N_c(a^{(\ell)}) \le n_b$ for all $\ell = 1, \ldots, q$. Therefore, the convolutions $a^{(\ell)} * b$ cost at most $\mathcal{O}(n_b \log n_b)$ for each ℓ. Finally, one has to add the results:

$$a * b = \sum_{\ell=0}^{q} a^{(\ell)} * b,$$

which requires $\mathcal{O}(n_a)$ additions. Here, we use that $N_c(a^{(\ell)} * b) = N_c(a^{(\ell)}) + N_c(b) - 1 \le \mathcal{O}(n_b)$ and $qn_b = n_a + n_b \le \mathcal{O}(n_a)$.

Altogether, the number of operations is $q * \mathcal{O}(n_b \log n_b) + \mathcal{O}(n_a) = \mathcal{O}(n_a \log n_b)$, since $q * \mathcal{O}(n_b) = \mathcal{O}(n_a)$. This proves the following Remark.

Remark 6.2 *The convolution $a * b$ of two finite sequences a and b of the respective sizes $N_c(a)$ and $N_c(b)$ can be performed with*

$$\mathcal{O}\left(\max\{N_c(a), N_c(b)\} \log\left(\min\{N_c(a), N_c(b)\}\right)\right)$$

operations.

The difference to the straightforward FFT approach with cost

$$\mathcal{O}(\max\{N_c(a), N_c(b)\} \log\left(\max\{N_c(a), N_c(b)\}\right))$$

is the minor change from $\max\{N_c(a), N_c(b)\}$ to $\min\{N_c(a), N_c(b)\}$ in the logarithmic factor. In particular, for $N_c(a) = \mathcal{O}(1)$ the logarithmic term vanishes because of $\log\left(\min\{N_c(a), N_c(b)\}\right) = \log(\mathcal{O}(1)) = \mathcal{O}(1)$.

6.6 Possible Cost Reductions for $L_c < N_c(c)$

As in §6.5 we assume (6.8a,b), i.e., a, b are sequences of the respective sizes $n_a = N_c(a)$, $n_b = N_c(b)$ with $S_c(a) = \{0, 1, \ldots, n_a - 1\}$ and $S_c(b) = \{0, 1, \ldots, n_b - 1\}$. Then the product $c = a * b$ has the support $S_c(c) = \{0, 1, \ldots, n_c - 1\}$ with $n_c = N_c(c) = N_c(a) + N_c(b) - 1$.

Now we assume that we are not interested in all non-zero components $(c_i)_{i=0}^{n_c-1}$ but only in those c_i with i from a set

$$\{\alpha_c, \alpha_c + 1, \ldots, \beta_c\} \subset S_c(c).$$

The length of the new interval $\{\alpha_c, \alpha_c + 1, \ldots, \beta_c\}$ is denoted by

$$L_c := \beta_c - \alpha_c + 1 \leq N_c(c).$$

Without loss of generality we assume $N_c(a) \leq N_c(b)$. The case $N_c(a) \leq N_c(b) \leq L_c$ is not of interest, since then $L_c = \mathcal{O}(N_c(c))$ is not really different from the unrestricted situation $L_c = N_c(c)$.

First we consider the case $N_c(a) \leq L_c \leq N_c(b)$ and claim that the required cost is

$$\mathcal{O}(L_c \log(N_c(a))).$$

Proof. Assume[9] $N_c(b) = mL_c$. We split b into m packages $b^{(1)}, \ldots, b^{(m)}$ of size $N_c(b^{(i)}) = L_c$. The support of $c^{(i)} := a * b^{(i)}$ has size $N_c(c^{(i)}) \leq N_c(a) + L_c + 1 \leq 2L_c + 1 = \mathcal{O}(L_c)$. The number of indices i with $S_c(c^{(i)}) \cap [\alpha_c, \beta_c] \neq \emptyset$ is a fixed number. Hence, in order to compute $c = \sum_i c^{(i)}$ restricted to $[\alpha_c, \beta_c]$, only $\mathcal{O}(1)$ convolutions $c^{(i)} := a * b^{(i)}$ are involved. The cost is $\mathcal{O}(1) \cdot \mathcal{O}(\max\{N_c(a), L_c\} \log(\min\{N_c(a), L_c\})) = \mathcal{O}(L_c \log(N_c(a)))$.

Next, we assume the case $L_c \leq N_c(a) \leq N_c(b)$. Now both a and b are split into packages of size L_c. The cost of $c^{(i,j)} := a^{(i)} * b^{(j)}$ is $\mathcal{O}(L_c \log(L_c))$. There are $\mathcal{O}(N_c(a)/L_c)$ combinations of i and j so that $S_c(c^{(i,j)}) \cap [\alpha_c, \beta_c] \neq \emptyset$. This leads to an arithmetical work of

$$\mathcal{O}(N_c(a) \log(L_c)).$$

Gathering the previous results and including the opposite inequality $N_c(a) \geq N_c(b)$, we obtain the following estimates.

Lemma 6.3 *For all sizes of $L_c, N_c(a)$, and $N_c(b)$ the restriction of $a*b$ to $[\alpha_c, \beta_c] \cap \mathbb{Z} \subset S_c(c)$ with $L_c = \beta_c - \alpha_c + 1 \leq N_c(c)$ can be computed such that the cost is*

$$\mathcal{O}\left(\max\left(L_c, \min\left(N_c(a), N_c(b)\right)\right) \log\left(\min\left(L_c, N_c(a), N_c(b)\right)\right)\right).$$

Corollary 6.4 *Since we have assumed that $L_c \leq \mathcal{O}(\max(N_c(a), N_c(b)))$, we can state the result of Lemma 6.3 also in the following form: The cost is bounded by $\mathcal{O}(A \log B)$, where $B := \min(L_c, N_c(a), N_c(b))$, while A is the second largest (also the second smallest) of the three quantities $L_c, N_c(a), N_c(b)$.*

[9] The case $N_c(b)/L_c \notin \mathbb{N}$ is treated as in §6.5 with one package of smaller size between 1 and $L_c - 1$.

For the practical computation, the sequences a, b should be shortened to sequences a', b' with possibly smaller support. This step can lead to a smaller amount of operations than considered in Lemma 6.3, because the intersection of the index set $\{\alpha_c, \alpha_c + 1, \ldots, \beta_c\}$ and the support of $a * b$ might be much smaller than in the worst case. Since

$$c_\nu := \sum_{\alpha \in \mathbb{Z}} a_\alpha b_{\nu-\alpha} = \sum_{\alpha=\max\{i_a, \nu-(i_b+N_c(b)-1)\}}^{\min\{i_a+N_c(a)-1, \nu-i_b\}} a_\alpha b_{\nu-\alpha}$$

for generally situated supports of a and b, the values c_ν with $\nu \in \{\alpha_c, \alpha_c+1, \ldots, \beta_c\}$ depend only on those components a_α with

$$\max\{i_a, \alpha_c - (i_b + N_c(b) - 1)\} \leq \alpha \leq \min\{i_a + N_c(a) - 1, \beta_c - i_b\}.$$

Let a' be the restriction to this index interval. Its length is

$$\begin{aligned} N_c(a') &= \min\{i_a + N_c(a) - 1, \beta_c - i_b\} - \max\{i_a, \alpha_c - (i_b + N_c(b) - 1)\} + 1 \\ &\leq \min\{N_c(a), L_c + N_c(b) - 1\}. \end{aligned} \tag{6.9}$$

Similarly, b can be reduced to b' with $N_c(b') \leq \min\{N_c(b), L_c + N_c(a') - 1\}$. In (6.9), $N_c(b)$ may be replaced by $N_c(b')$.

6.7 Possible Cost Reductions for $N_d(a) \ll N_c(a)$

We recall that the cardinality $N_d(a)$ of the discrete support is the true number of non-zero entries of a, whereas $N_c(a)$ is the size after convexification. The difference $N_c(a) - N_d(a)$ is the number of zero components in $S_c(a)$.

Figure 6.1: The basis grid (upper line) is refined at the endpoints. Formally, the gap between contains zero components.

A typical and realistic example is shown in Figure 6.1. A first uniform grid is refined close to both endpoints[10]. The grid points of the left and right fine meshes correspond to the discrete support $S_d(a)$. The gap in the middle leads to zero values $a_\nu = 0$. Depending of the size of the gap, $N_d(a)$ may be much smaller than $N_c(a)$. Unfortunately, the cost of computing $\hat{a}$ is described by $N_c(a)$, since, in general, $\hat{a}$ has no extra zero components.

Let $a * b$ to be computed. If $N_c(a) \leq N_c(b) \approx N_d(b)$, the computational cost is mainly determined by $N_c(b)$ (see Remark 6.2). In this case, improvements exploiting $N_d(a) \ll N_c(a)$ are less effective. Therefore, we next suppose the case $N_d(a) \approx N_c(b) \ll N_c(a)$.

In the case of *one* big gap as in Figure 6.1, i.e., $a_\nu = 0$ for $n' \leq \nu < n''$, we can split the sequence $a = (a_\nu)_{\nu=0}^{n_a-1}$ into

$$a = a' + a'', \quad \text{where } a' = (a_\nu)_{\nu=0}^{n'-1}, \; a'' = (a_\nu)_{\nu=n''}^{n_a-1}. \tag{6.10}$$

[10]Refinement at only one endpoint leads to the optimal case of $N_c(a) = N_d(a)$.

Since $N_c(a') = n'$ and $N_c(a'') = n_a - n''$, we have

$$N_c(a') + N_c(a'') = N_c(a) - g \quad \text{with the gap-length } g := n'' - n'.$$

Note that in the case that all components a_ν for $\nu \in [0, n'-1] \cup [n'', n_a - 1]$ are non-zero, we have $N_c(a') + N_c(a'') = N_d(a)$.

Instead of $a * b$ we perform $a' * b$ and $a'' * b$ and form the sum. The overall cost is bounded by

$$\mathcal{O}\left(\max\{N_c(a') + N_c(a''), N_c(b)\} \log\left(\min\{N_c(a') + N_c(a''), N_c(b)\}\right)\right).$$

Provided that $N_c(a') + N_c(a'') \approx N_d(a)$, we obtain the cost

$$\mathcal{O}\left(\max\{N_d(a), N_c(b)\} \log\left(\min\{N_d(a), N_c(b)\}\right)\right).$$

If the assumptions apply to a *and* b, even the cost

$$\mathcal{O}\left(\max\{N_d(a), N_d(b)\} \log\left(\min\{N_d(a), N_d(b)\}\right)\right)$$

can be reached. These numbers may improve when we make use of Lemma 6.3.

In the situation that there are two gaps, one can try to split a into $a = a' + a'' + a'''$, but now more sequences $a' * b, \ldots$ must be added.

The most unfavourable case is given by examples where the zero components are spread almost equally among the non-zero components. Then the gap lengths are rather small and the construction from above does no work.

Nevertheless, even for more than one gap there is hope that $N_d(a) \ll N_c(a)$ or $N_d(b) \ll N_c(b)$ can be exploited. In §6.6 we have seen that either a or b or both must be split into many packages $a^{(i)}$ or $b^{(j)}$ of smaller length. In the situation of (6.10) it is very likely that the first package $a^{(1)}$ equals a', the last package equals a'' and all other ones meet the gap, i.e., $a^{(i)} = 0$. Omitting the latter zero-sequences, we have optimally exploited the fact that $N_d(a) \ll N_c(a)$.

7 Analysis of the Computational Costs

7.1 Recursion Formulae

(5.5) and (A.4a,b) are typical examples where sequences on different levels are determined via[11]

$$a_i^{(\ell)} := \tfrac{1}{\sqrt{2}}\left(a_{2i}^{(\ell+1)} + a_{2i+1}^{(\ell+1)}\right) \qquad \text{for all } i \in \mathbb{Z}. \tag{7.1}$$

Such a recursion starts with a given sequence $a^{(L)}$ and is performed for $\ell = L-1, L-2, \ldots, 0$.

Let $a^{(L)}$ have the length $N := N_c(a^{(L)})$. The length of $a^{(L-1)}$ is $\frac{N}{2}$ for even N and $\frac{N+1}{2}$ for odd N. Hence,

$$N_c(a^{(\ell)}) \leq \left\lfloor \frac{N_c(a^{(\ell+1)}) + 1}{2} \right\rfloor \qquad \text{for all } 0 \leq \ell \leq L-1, \tag{7.2}$$

where the rounding $\lfloor x \rfloor$ is the largest integer $\leq x$.

[11]The kind of linear combination does not matter.

Lemma 7.1 *The inequalities (7.2) lead to*

$$N_c(a^{(\ell)}) \leq 2^{\ell-L} N_c(a^{(L)})) + 1 - 2^{\ell-L} \qquad \text{for } 0 \leq \ell \leq L.$$

The size is almost geometrically decreasing. Performance of (7.1) for all $\ell = L-1$, $L-2, \ldots, 0$ (of course restricted to the non-zero components of $a_i^{(\ell)}$) requires $\mathcal{O}(N+L)$ operations (more precisely, $2N + L - 1$ for the particular linear combination in (7.1)). Since usually $L \leq N$, we may simplify the complexity to $\mathcal{O}(N) = \mathcal{O}(N_c(a^{(L)}))$.

The performance of (7.1) requires two operations for each component $a_i^{(\ell)}$, hence $2N_c(a^{(\ell)})$ for the whole sequence. The constant 1 in $N_c(a^{(\ell)}) \leq 2^{\ell-L} N_c(a^{(L)})) + 1$ seems harmless, however, in the appendix we may have $2^{L-\ell}$ problems of the size $N_c(a^{(\ell)})$, so that the overall number of arithmetical operations is proportional to $2^{L-\ell} N_c(a^{(\ell)}) \leq N_c(a^{(L)})) + 2^{L-\ell}$. In this case, the exponential growth with respect to the level number L is undesirable.

The remedy of this problem is discussed in the following two cases.
Case 1: $N_c(a^{(L)}) \geq 2^L$. Then $1 \leq 2^{\ell-L} N_c(a^{(L)})$, so that

$$N_c(a^{(\ell)}) \leq 2^{\ell-L} N_c(a^{(L)})) + 1 \leq 2 \cdot 2^{\ell-L} N_c(a^{(L)}) \qquad \text{for all } 0 \leq \ell \leq L.$$

Case 2: $N_c(a^{(L)}) < 2^L$. Then there is a level $\ell_1 \in [1, L] \cap \mathbb{N}$ with $1 \leq 2^{\ell-L} N_c(a^{(L)}))$ for all $\ell_1 < \ell \leq L$ and $1 > 2^{\ell-L} N_c(a^{(L)})$ for all $0 \leq \ell \leq \ell_1$. For $\ell_1 < \ell \leq L$ we have again $N_c(a^{(\ell)}) \leq 2 \cdot 2^{\ell-L} N_c(a^{(L)})$, whereas for $0 \leq \ell \leq \ell_1$ the identity $N_c(a^{(\ell)}) = 1$ follows from $N_c(a^{(\ell)}) \leq 2^{\ell-L} N_c(a^{(L)}) + 1 - 2^{\ell-L} < 2$ and the fact that $N_c(a^{(\ell)}) = 1$ implies $N_c(a^{(\ell-1)}) = 1$. In the latter case we use the algorithm as described below.

Remark 7.2 *a) As soon as the recursion (7.1) leads to a sequence $a^{(\ell_1)}$ with only one non-zero component (i.e., $N_c(a^{(\ell_1)}) = 1$), the recursion (7.1) is no more used for the numerical computation. Instead we know that $a_{i_1}^{(\ell_1)} \neq 0$ implies $a_{i(\ell)}^{(\ell)} = a_{i_1}^{(\ell_1)} / 2^{(\ell_1-\ell)/2}$ for $i(\ell) = \lfloor i_1 / 2^{\ell_1-\ell} \rfloor$ and $a_i^{(\ell)} = 0$ for all other $i \in \mathbb{Z}$.*

b) In particular, a discrete convolution $a^{(\ell)} * b$ yields the sequence c with $c_i = \frac{a_{i_1}^{(\ell_1)}}{2^{(\ell_1-\ell)/2}} b_{i-i(\ell)}$. This involves a shift and multiplication by a constant. The cost is, in general, $N_c(b)$, and if only L_c components are wanted, $\min\{L_c, N_c(b)\}$.

c) In Case 1 and in the first subcase of Case 2 from above, the convolution $a^{(\ell)} * b$ (restricted to $L_c \leq N_c(a^{(\ell)}) + N_c(b)$ components) requires

$$\begin{aligned}
&\mathcal{O}\left(\max\left(L_c, \min\left(N_c(a^{(\ell)}), N_c(b)\right)\right) \log\left(\min\left(L_c, N_c(a^{(\ell)}), N_c(b)\right)\right)\right) \\
&\leq \mathcal{O}\left(\max\left(L_c, \min\left(2^{\ell-L} N_c(a^{(L)}), N_c(b)\right)\right) \log\left(\min\left(L_c, 2^{\ell-L} N_c(a^{(L)}), N_c(b)\right)\right)\right)
\end{aligned}$$

operations. In the second subcase of Case 2, the cost $\min\{L_c, N_c(b)\}$ from Part b) can be estimated by the same bound:

$$\begin{aligned}
&\min\{L_c, N_c(b)\} \leq L_c \\
&\leq \mathcal{O}\left(\max\left(L_c, \min\left(2^{\ell-L} N_c(a^{(L)}), N_c(b)\right)\right) \log\left(\min\left(L_c, 2^{\ell-L} N_c(a^{(L)}), N_c(b)\right)\right)\right).
\end{aligned}$$

7.2 Case A

7.2.1 Step 1

Algorithm (5.7) corresponds to the Steps 1a,b in §5.1.1.

The computation of $\Gamma_i^{\ell,\ell}$ in (5.2) costs $\mathcal{O}(N_c(g_\ell))$ operations. In line 4 of (5.7) we need $\Gamma_i^{\ell',\ell'}$ for all $0 \le \ell' \le L$. This requires

$$\mathcal{O}\left(N_c(g)\right)$$

operations. This includes the addition $\Gamma_i^{\ell'} + \Gamma_i^{\ell',\ell'}$.

The recursion (5.3) is not exactly of the form (7.1), but the cost for its performance is similar. This proves the following statement.

The computation of $\Gamma_i^{\ell',\ell}$ for all $\ell' = \ell-1, \ell-2, \ldots, 0$ costs $\mathcal{O}(N_c(g_\ell))$ operations. Lines 3-4 in (5.7) refer to $\Gamma_i^{\ell'}$ which is the sum of $\Gamma_i^{\ell',\ell}$ over all $\ell \ge \ell'$. This fact increases $N_c(g_\ell)$ to $N_{cc}(g_\ell)$, so that the total work is

$$\mathcal{O}\left(N_{cc}(g)\right). \tag{7.3a}$$

7.2.2 Step 2

The recursion in line 3 of (5.8) has to be done for the true support of

$$\omega_{\ell''} = \sum_{\ell'=\ell''}^{L} P_{\ell''}(f_{\ell'} * \Gamma_{\ell'}).$$

The number of coefficients is bounded by $\mathcal{O}\left(\sum_{\ell'=\ell''}^{L} 2^{\ell''-\ell'}\left(N_c(f_{\ell'}) + N_c(\Gamma_{\ell'})\right)\right)$. Summation over $0 \le \ell'' \le L$ yields the bound $\mathcal{O}\left(\sum_{\ell'=0}^{L}\left(N_c(f_{\ell'}) + N_c(\Gamma_{\ell'})\right)\right) = \mathcal{O}\left(N_c(f) + \sum_{\ell'=0}^{L} N_c(\Gamma_{\ell'})\right)$. As in the previous step, we have $\mathcal{O}\left(\sum_{\ell'=0}^{L} N_c(\Gamma_{\ell'})\right) \le \mathcal{O}\left(\sum_{\ell'=0}^{L} N_{cc}(g_{\ell'})\right) = \mathcal{O}\left(N_{cc}(g)\right)$ as in (7.3a).

In line 4 of (5.8), the discrete convolution $\omega_{\ell''} := f_{\ell''} * \Gamma_{\ell''}$ from (5.4) is to be evaluated for all $0 \le \ell'' \le L$. The length $N_c(\Gamma_{\ell'})$ equals $\mathcal{O}(N_{cc}(g_{\ell'}))$. Hence, the convolution cost is $\mathcal{O}(\max\{N_c(f_{\ell''}), N_c(\Gamma_{\ell''})\}\log(\min\{N_c(f_{\ell''}), N_c(\Gamma_{\ell''})\}))$. Summation over all ℓ'' yields the bound

$$\mathcal{O}(\max\{N_c(f), N_{cc}(g)\}\log(\min\{N_c(f), N_{cc}(g))\}). \tag{7.3b}$$

7.3 Case B

We analyse the cost of (5.11). The number of coefficients involved in line 4 of (5.11) is $N_{cc}(\omega_{\ell''}) + N_c(\Gamma_{\ell''})$. Since $\sum_{\ell''=1}^{L} N_{cc}(\omega_{\ell''}) \le \mathcal{O}(\sum_{\ell''=1}^{L} N_c(\omega_{\ell''})) = \mathcal{O}(N_c(\omega))$ (see (6.3)), this cost is less than the next part.

The convolution in line 5 costs

$$\begin{aligned}&\mathcal{O}\left(\max\left(N_c(\omega_{\ell''}), \min\{N_c(\hat{f}_{\ell''}), N_c(\Gamma_{\ell''})\}\right)\log\left(\min\{N_c(\omega_{\ell''}), N_c(\hat{f}_{\ell''}), N_c(\Gamma_{\ell''})\}\right)\right)\\ &\le \mathcal{O}\left(\max\left(N_c(\omega_{\ell''}), N_c(\Gamma_{\ell''})\right)\log\left(\min\{N_c(\omega), N_c(g)\}\right)\right).\end{aligned}$$

Summation over all $1 \le \ell'' \le L$ yields the upper bound

$$\mathcal{O}\left(\max\left(N_c(\omega), N_c(g)\right)\log\left(\min\{N_c(\omega), N_c(g)\}\right)\right) \tag{7.4}$$

for the cost of algorithm (5.11).

7.4 Case C

The components $\hat{\delta}_i^\ell$ are needed for all $\omega_{\ell''}$ with $\ell'' > \ell$. Therefore their number is $\mathcal{O}(N_{cc}(\omega_{\ell+1}))$. To obtain these δ_i^ℓ-values by the convolution $\sum_{j\in\mathbb{Z}} \hat{f}_j^\ell g_{i-j-1}^\ell$, the number of necessary components $\hat{f}_i^\ell$ is $\mathcal{O}(N_{cc}(\omega_{\ell+1}) + N_c(g_\ell))$. Hence, the lines 3-6 and 9-11 of algorithm (5.16) lead to $\mathcal{O}(N_{cc}(\omega_1) + N_c(g))$ operations, while the convolution in line 8 takes $\mathcal{O}\left(\max\left\{N_{cc}(\omega_{\ell+1}), N_c(g_\ell)\right\}\log\left(\min\left\{N_{cc}(\omega_{\ell+1}), N_c(g_\ell)\right\}\right)\right)$ operations for each ℓ. The total sum is bounded by

$$\mathcal{O}\left(\left(N_{cc}(\omega) + N_c(g)\right)\log\left(N_c(g)\right)\right). \tag{7.5}$$

7.5 Overall Costs

Adding (7.3a,b) for Case A, (7.4) for Case B, and (7.5) for Case C, we obtain the final result for the cost:

$$\mathcal{O}(\max\left\{(N_{cc}(\omega) + N_c(f) + N_{cc}(g))\right\}\log\left(\min\left\{N_c(f) + N_c(g))\right\}\right). \tag{7.6}$$

Under the condition (6.5), this bound becomes

$$\mathcal{O}(N\log N), \qquad \text{where } N := N_c(\omega) + N_c(f) + N_c(g).$$

N describes the total data size of the three function $f \in \mathcal{S}^f,\ g \in \mathcal{S}^g,\ \omega \in \mathcal{S}^\omega$.

8 General Polynomial Ansatz Functions

The purpose of this chapter is to explain why the described method can be generalised to higher polynomial degrees. Details will be given a forthcoming paper. The example of piecewise linear ansatz functions can be found in [3].

8.1 Basis Functions

Now we allow that the function f restricted to $I_i^\ell \in \mathcal{M}_\ell$ is represented by a polynomial of degree p_i^ℓ (see §2.2). In this case we need $p_i^\ell + 1$ basis functions per interval. For this purpose we replace the index i used for the piecewise constant basis function Φ_i^ℓ by the pair (i,α), where $i \in \mathbb{Z}$ refers to the interval, while $\alpha \in \{0, 1, \ldots, p_i^\ell\}$ distinguishes the basis functions $\Phi_{(i,\alpha)}^\ell$ with support in I_i^ℓ. As before, we need *orthonormal* basis functions $\{\Phi_{(i,\alpha)}^\ell : i \in \mathbb{Z},\ 0 \le \alpha \le p_i^\ell\}$. The best choice are the *Legendre polynomials*; more precisely, the standard Legendre polynomials of degree α defined in the reference interval $[-1, 1]$ are to be mapped onto I_i^ℓ by an affine mapping and scaled such that $\int |\Phi_{(i,\alpha)}^\ell|^2 \mathrm{d}x = 1$. In the following, the choice of the Legendre polynomials as basis functions is required[12].

[12]Other orthonormal sets of basis functions are possible, but the Legendre polynomials are the only ones with the property that the degree of $\Phi_{(i,\alpha)}^\ell$ is α.

8.2 γ-Coefficients

Replacing i, j, k by pairs (i, α), (j, β), $(k, \varkappa)$, we define analogously to (4.1)

$$\gamma^{\ell'',\ell',\ell}_{(i,\alpha),(j,\beta),(k,\varkappa)} := \iint \Phi^{\ell''}_{(i,\alpha)}(x)\, \Phi^{\ell'}_{(j,\beta)}(y)\, \Phi^{\ell}_{(k,\varkappa)}(x-y)\mathrm{d}x\mathrm{d}y. \tag{8.1}$$

Again, simplified γ-coefficients can be defined by setting $i = j = 0$, however their simplicity is reduced by the fact that they depend on the triple $\alpha, \beta, \varkappa$:

$$\gamma^{\ell'',\ell',\ell}_{\nu,(\alpha,\beta,\varkappa)} := \iint \Phi^{\ell''}_{(0,\alpha)}(x)\, \Phi^{\ell'}_{(0,\beta)}(y)\, \Phi^{\ell}_{(\nu,\varkappa)}(x-y)\mathrm{d}x\mathrm{d}y.$$

As in Lemma 4.2 we can prove the following relation.

Lemma 8.1 *Let $\ell \geq \max\{\ell', \ell''\}$. Then*

$$\gamma^{\ell'',\ell',\ell}_{(i,\alpha),(j,\beta),(k,\varkappa)} = \gamma^{\ell'',\ell',\ell}_{k-i2^{\ell-\ell''}+j2^{\ell-\ell'},(\alpha,\beta,\varkappa)} \qquad \textit{for any } i, j, k \in \mathbb{Z}\,. \tag{8.2}$$

Some of these coefficients need not be computed since they vanish.

Lemma 8.2 $\gamma^{\ell'',\ell',\ell}_{\nu,(\alpha,\beta,\varkappa)} = 0$ *for all $\nu \in \mathbb{Z}$ follows from each of the following conditions:*
a) $\ell'' \geq \max\{\ell, \ell'\}$ and $\alpha > \beta + \varkappa + 1$,
b) $\ell' \geq \max\{\ell, \ell''\}$ and $\beta > \alpha + \varkappa + 1$,
c) $\ell \geq \max\{\ell', \ell''\}$ and $\varkappa > \alpha + \beta + 1$.

Proof. a) Since $\Phi^{\ell'}_{(0,\beta)}$ and $\Phi^{\ell}_{(\nu,\varkappa)}$ are of degree β and $\varkappa$, the convolution $\Phi^{\ell'}_{(0,\beta)} * \Phi^{\ell}_{(\nu,\varkappa)}$ is locally – in particular, in any support of $\Phi^{\ell''}_{(0,\alpha)}$ – a polynomial of degree $\leq \beta + \varkappa + 1$. Because $\Phi^{\ell''}_{(0,\alpha)}$ is orthogonal to any polynomial of degree smaller than α, we get $\gamma^{\ell'',\ell',\ell}_{\nu,(\alpha,\beta,\varkappa)} = \left\langle \Phi^{\ell''}_{(0,\alpha)}, \Phi^{\ell'}_{(0,\beta)} * \Phi^{\ell}_{(\nu,\varkappa)} \right\rangle = 0$.

b) Define $\check{\Phi}^{\ell}_{(\nu,\varkappa)}(x) := \Phi^{\ell}_{(\nu,\varkappa)}(-x)$. Then $\int \Phi^{\ell''}_{(0,\alpha)}(x)\, \Phi^{\ell}_{(\nu,\varkappa)}(x-y)\mathrm{d}x = \left(\Phi^{\ell''}_{(0,\alpha)} * \check{\Phi}^{\ell}_{(\nu,\varkappa)}\right)(y)$ is locally a polynomial of degree $\leq \alpha + \varkappa + 1$. Since $\gamma^{\ell'',\ell',\ell}_{\nu,(\alpha,\beta,\varkappa)} = \left\langle \Phi^{\ell'}_{(0,\beta)}, \Phi^{\ell''}_{(0,\alpha)} * \check{\Phi}^{\ell}_{(\nu,\varkappa)} \right\rangle$, we argue as in part a).

c) The last case follows from the symmetry $\Phi^{\ell'}_{(0,\beta)} * \Phi^{\ell}_{(\nu,\varkappa)} = \Phi^{\ell}_{(\nu,\varkappa)} * \Phi^{\ell'}_{(0,\beta)}$. ■

8.3 Recursion Formulae

The support of $\Phi^{\ell}_{(i,\varkappa)}$ is the interval $I^{\ell}_{i} = [ih_\ell, (i+1)\,h_\ell)$. Its restriction $\Phi^{\ell}_{(i,\varkappa)}|_{I^{\ell+1}_{2i}}$ to the first subinterval $I^{\ell+1}_{2i} = [ih_\ell, (i+1/2)\,h_\ell)$ is a polynomial of degree $\varkappa$ and can therefore be written as a linear combination of the basis functions $\Phi^{\ell+1}_{(2i,\alpha)}$ $(0 \leq \alpha \leq \varkappa)$ in $I^{\ell+1}_{2i}$:

$$\Phi^{\ell}_{(i,\varkappa)}(x) = \sum_{\alpha=0}^{\varkappa} \xi_{\varkappa,\alpha} \Phi^{\ell+1}_{(2i,\alpha)}(x) \qquad \text{for } x \in I^{\ell+1}_{2i} = [ih_\ell, (i+1/2)\,h_\ell)\,.$$

Remark 8.3 *a) By symmetry reasons the representation in the other part $I_{2i+1}^{\ell+1}$ is*

$$\Phi_{(i,\varkappa)}^{\ell}(x) = \sum_{\alpha=0}^{\varkappa} (-1)^{\alpha+\varkappa} \xi_{\varkappa,\alpha} \Phi_{(2i+1,\alpha)}^{\ell+1}(x) \qquad \text{for } x \in I_{2i+1}^{\ell+1} = [(i+1/2)\, h_\ell, (i+1)\, h_\ell)\,,$$

so that altogether we obtain the representation

$$\Phi_{(i,\varkappa)}^{\ell} = \sum_{\alpha=0}^{\varkappa} \xi_{\varkappa,\alpha} \left(\Phi_{(2i,\alpha)}^{\ell+1} + (-1)^{\alpha+\varkappa} \Phi_{(2i+1,\alpha)}^{\ell+1} \right). \tag{8.3}$$

b) The coefficients $\xi_{\varkappa,\alpha}$ depend neither on i nor on ℓ.

The identity (8.3) is the generalisation of (2.8), where $\xi_{0,0} = 1/\sqrt{2}$.

The representations of $f = \sum_{\ell'} f_{\ell'}$ and $g = \sum_{\ell} g_\ell$ require a further sum over all degrees:

$$f_{\ell'} = \sum_{j \in \mathcal{I}_{\ell'}^{f}} \sum_{\beta=0}^{p_j^{f,\ell'}} f_{(j,\beta)}^{\ell'} \Phi_{(j,\beta)}^{\ell'} \in \mathcal{S}_{\ell'}^{f}, \qquad g_{\ell'} = \sum_{k \in \mathcal{I}_{\ell}^{g}} \sum_{\varkappa=0}^{p_j^{g,\ell}} g_{(k,\varkappa)}^{\ell} \Phi_{(k,\varkappa)}^{\ell'} \in \mathcal{S}_{\ell}^{g} \tag{8.4}$$

(cf. (3.2), (3.3)). The local polynomial degrees $p_j^{f,\ell'}$ and $p_j^{g,\ell}$ are part of the new definition of $\mathcal{S}_{\ell'}^{f}$ and $\mathcal{S}_{\ell}^{g}$. Similarly, $\mathcal{S}_{\ell}^{\omega}$ involves polynomial degrees $p_i^{\omega,\ell''}$. For convenience, we replace the individual degrees $p_j^{f,\ell'}$, $p_j^{g,\ell}$, $p_i^{\omega,\ell''}$ by their maximum p.

The projected convolution takes the form $\omega = \sum_{\ell''} \omega_{\ell''}$ with $\omega_{\ell''} = \sum_{i \in \mathcal{I}_{\ell''}^{\omega}} \sum_{\alpha=0}^{p} \omega_{(i,\alpha)}^{\ell''} \Phi_{(i,\alpha)}^{\ell'}$. The coefficients satisfy

$$\omega_{(i,\alpha)}^{\ell''} = \sum_{(j,\beta)} \sum_{(k,\varkappa)} f_{(j,\beta)}^{\ell'} g_{(k,\varkappa)}^{\ell} \gamma_{(i,\alpha),(j,\beta),(k,\varkappa)}^{\ell'',\ell',\ell}$$

(cf. (4.3)).

Again summation over k and $\varkappa$ defines the coefficients

$$G_{(i,\alpha),(j,\beta)}^{\ell'',\ell',\ell} = \sum_{\varkappa=0}^{p} \sum_{k \in \mathbb{Z}} g_{(k,\varkappa)}^{\ell} \gamma_{(i,\alpha),(j,\beta),(k,\varkappa)}^{\ell'',\ell',\ell}$$

and allows to obtain $\omega_{\ell''}$ from

$$\omega_{(i,\alpha)}^{\ell''} = \sum_{\beta=0}^{p} \sum_{j \in \mathbb{Z}} f_{(j,\beta)}^{\ell'} G_{(i,\alpha),(j,\beta)}^{\ell'',\ell',\ell}.$$

The only difference to the piecewise constant case is (a) the additional sum the degree index (β or $\varkappa$) and (b) other recursion formulae. For instance, the recursion $G_{i,j}^{\ell'',\ell',\ell} = \frac{1}{\sqrt{2}} \left(G_{2i,j}^{\ell''+1,\ell',\ell} + G_{2i+1,j}^{\ell''+1,\ell',\ell} \right)$ from (4.10a) now becomes

$$G_{(i,\alpha),(j,\beta)}^{\ell'',\ell',\ell} = \sum_{\delta=0}^{\alpha} \xi_{\alpha,\delta} \left(G_{(2i,\delta),(j,\beta)}^{\ell''+1,\ell',\ell} + (-1)^{\delta+\alpha} G_{(2i+1,\delta),(j,\beta)}^{\ell''+1,\ell',\ell} \right).$$

8.4 Algorithm, Cost

The generalisation of the algorithm is straightforward. Even the estimation of the cost follows the same lines provided that the maximal degree p can be considered as a constant.

9 Convolution of Periodic Functions

Let 2π be the period of the functions f and g. Then the corresponding *periodic convolution* is defined by

$$\omega_{\text{exact}}(x) := (f * g)(x) := \int_0^{2\pi} f(y)g(x-y)\mathrm{d}y \qquad \text{for all } x \in [0, 2\pi)$$

instead of (1.1). The computation of the projected periodic convolution is very similar to the convolution in $\mathbb{R}$. We only list the items which are to be changed.

The step size h of the coarsest grid must be restricted to the discrete set $\{2\pi/m : m \in \mathbb{N}\}$. In the following we assume that m denotes the integer such that $h = h_0 = 2\pi/m$.

The set of basis functions is now finite. The index i of Φ_i^ℓ is restricted to the set $\{0, 1, \ldots, 2^\ell m - 1\}$. Furthermore, the location index i must be understood modulo $2^\ell m$. This holds in particular for the index $\nu - \alpha$ in the following *discrete periodic convolution* (of period $2^\ell m$):

$$c := a * b, \text{ i.e.,} \qquad c_\nu := \sum_{\alpha=0}^{2^\ell m - 1} a_\alpha b_{\nu-\alpha} \quad \text{for all } 0 \le \nu \le 2^\ell m - 1.$$

Also intervals in $\mathbb{Z}/\left(2^\ell m\mathbb{Z}\right)$ are to be understood in the period sense. If

$$0 \le i_{\min} \le i_{\max} < 2^\ell m,$$

$[i_{\min}, i_{\max}]$ is a usual interval embedded into $\mathbb{Z}_{2^\ell m}$ containing $i_{\max} - i_{\min} + 1$ elements. In the case $0 \le i_{\max} < i_{\min} < 2^\ell m$, the union $\{i_{\min}, i_{\min}+1, \ldots, 2^\ell m - 1\} \cup \{0, 1, \ldots, i_{\max}\}$ of two standard intervals can be interpreted as one interval $\{i_{\min} - 2^\ell m, i_{\min} - 2^\ell m + 1, \ldots, i_{\max}\}$ modulo $2^\ell m$ containing $i_{\max} - i_{\min} + 2^\ell m + 1$ elements. This fact is important for a new definition of the convexified set $S_c(\cdot)$. We define $S_c(\cdot)$ as the smallest interval (in the periodic sense) containing the set $S_d(\cdot)$. In the example depicted in Figure 6.1, the grid at level 1 leads to $S_c(\cdot) = S_d(\cdot)$!

For the basic discrete convolution problem (see §6.2) we can always use the fixed value[13] $2^\ell m$ (depending on the level number ℓ, but not depending on n). The former condition $2^\ell m \ge 2n - 1$ is not required. However, if $2^\ell m \ge 2n - 1$ holds (and hopefully there are many discrete convolution problems with rather small n), the discrete convolution can be performed as be in the non-period fashion. Therefore, the estimates of the cost are the same as before.

[13] $2^\ell m$ replaces m in §6.2.

References

[1] M.V. Fedorov, H.-J. Flad, G.N. Chuev, L. Grasedyck, and B.N. Khoromskij: *Low-rank wavelet solver for the Ornstein-Zernike integral equation*. Max-Planck-Institut für Mathematik in den Naturwissenschaften. Preprint 59/2005, Leipzig 2005 (to appear in Computing).

[2] W. Hackbusch: *On the efficient evaluation of coalescence integrals in population balance models*. Computing **78** (2006) 145-172.

[3] W. Hackbusch: *Fast and exact projected convolution of piecewise linear functions on non-equidistant grids*. Preprint 110, Max-Planck-Institut für Mathematik, Leipzig, 2006.

[4] D. Potts: *Schnelle Fourier-Transformationen für nichtäquidistante Daten und Anwendungen*. Habilitation thesis, Universität zu Lübeck, 2003.

[5] D. Potts, G. Steidl, and A. Nieslony: *Fast convolution with radial kernels at nonequispaced knots*. Numer. Math. **98** (2004) 329-351.

In the Sections of the appendix we give alternative approaches to the Cases A-C, which however need more work than the algorithms given in the main part.

A Appendix: Second Variant of Step 2b in Case A

A.1 Algorithm

In §5.1.3, the convolution is first evaluated at level ℓ' and then coarsened to levels $\ell'' \le \ell'$. Therefore, $\omega_{\ell'}$ has to be computed in its whole support $S_c(\omega_{\ell'})$. The following variant tries to compute $\omega_{\ell'}$, $\omega_{\ell'-1}$, ... individually, so that their evaluation requires only the necessary part. However, it will turn out that the costs are higher than for the variant presented in §5.1.3.

The second variant uses another organisation of the j-sum in $\omega_i^{\ell''} = \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \gamma_{i,j,k}^{\ell'',\ell',\ell}$. First, we give the example $\ell'' = \ell' - 1$ in detail, before we describe the general procedure.

A.1.1 Example $\ell'' = \ell' - 1$

We split the sequence $f_{\ell'} = \left(f_j^{\ell'}\right)_{j\in\mathbb{Z}}$ into two subsequences $f_{\ell',0} = \left(f_{2j}^{\ell'}\right)_{j\in\mathbb{Z}}$ and $f_{\ell',1} = \left(f_{2j+1}^{\ell'}\right)_{j\in\mathbb{Z}}$. Then the summation over j in $\omega_i^{\ell'-1} = \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \gamma_{i,j,k}^{\ell'-1,\ell',\ell}$ becomes

$$\omega_i^{\ell'-1} = \sum_{j,k\in\mathbb{Z}} f_{2j}^{\ell'} g_k^{\ell} \gamma_{i,2j,k}^{\ell'-1,\ell',\ell} + \sum_{j,k\in\mathbb{Z}} f_{2j+1}^{\ell'} g_k^{\ell} \gamma_{i,2j+1,k}^{\ell'-1,\ell',\ell}.$$

For the first γ-factor we use

$$\gamma_{i,2j,k}^{\ell'-1,\ell',\ell} \underset{(4.6)}{=} \gamma_{k-i2^{\ell-(\ell'-1)}+(2j)2^{\ell-\ell'}}^{\ell'-1,\ell',\ell} = \gamma_{k-(i-j)2^{\ell-(\ell'-1)}}^{\ell'-1,\ell',\ell} \underset{(4.6)}{=} \gamma_{i-j,0,k}^{\ell'-1,\ell',\ell}.$$

The second γ-factor is treated similarly:

$$\gamma_{i,2j+1,k}^{\ell'-1,\ell',\ell} = \gamma_{k-i2^{\ell-(\ell'-1)}+(2j+1)2^{\ell-\ell'}}^{\ell'-1,\ell',\ell} = \gamma_{k-(i-j)2^{\ell-(\ell'-1)}+2^{\ell-\ell'}}^{\ell'-1,\ell',\ell} = \gamma_{i-j,1,k}^{\ell'-1,\ell',\ell}.$$

Summation over k yields

$$\sum_{k\in\mathbb{Z}} g_k^{\ell} \gamma_{i,2j,k}^{\ell'-1,\ell',\ell} = \sum_{k\in\mathbb{Z}} g_k^{\ell} \gamma_{i-j,0,k}^{\ell'-1,\ell',\ell} = G_{i-j,0}^{\ell'-1,\ell',\ell}$$

and $\sum_{k\in\mathbb{Z}} g_k^{\ell} \gamma_{i,2j+1,k}^{\ell'-1,\ell',\ell} = G_{i-j,1}^{\ell'-1,\ell',\ell}$. The final result is

$$\omega_i^{\ell'-1} = \sum_{j\in\mathbb{Z}} f_{2j}^{\ell'} G_{i-j,0}^{\ell'-1,\ell',\ell} + \sum_{j\in\mathbb{Z}} f_{2j+1}^{\ell'} G_{i-j,1}^{\ell'-1,\ell',\ell}.$$

Hence, introducing the notations $G_{\nu}^{\ell'-1,\ell',\ell} := \left(G_{i,\nu}^{\ell'-1,\ell',\ell}\right)_{i\in\mathbb{Z}}$ with $\nu \in \{0,1\}$, we can represent the sequence $\omega_{\ell'-1} = \left(\omega_i^{\ell'-1}\right)_{i\in\mathbb{Z}}$ by the sum of two discrete convolutions:

$$\omega_{\ell'-1} = f_{\ell',0} * G_0^{\ell'-1,\ell',\ell} + f_{\ell',1} * G_1^{\ell'-1,\ell',\ell}.$$

In order to get the coefficients of $G_0^{\ell'-1,\ell',\ell}$, $G_1^{\ell'-1,\ell',\ell}$. We use (4.10a) and (4.11):

$$G_{i,0}^{\ell'-1,\ell',\ell} = \frac{1}{\sqrt{2}}\left(G_{2i,0}^{\ell',\ell',\ell} + G_{2i+1,0}^{\ell',\ell',\ell}\right) = \frac{1}{\sqrt{2}}\left(\Gamma_{2i}^{\ell',\ell} + \Gamma_{2i+1}^{\ell',\ell}\right), \tag{A.1}$$

$$G_{i,1}^{\ell'-1,\ell',\ell} = \frac{1}{\sqrt{2}}\left(G_{2i,1}^{\ell',\ell',\ell} + G_{2i+1,1}^{\ell',\ell',\ell}\right) \underset{(4.9)}{=} \frac{1}{\sqrt{2}}\left(G_{2i-1,0}^{\ell',\ell',\ell} + G_{2i,0}^{\ell',\ell',\ell}\right) = \frac{1}{\sqrt{2}}\left(\Gamma_{2i-1}^{\ell',\ell} + \Gamma_{2i}^{\ell',\ell}\right).$$

Note that the coefficients $\Gamma_i^{\ell',\ell}$ are known from Step 1b (see §5.1.1).

A.1.2 General Recursion

Let $\ell'' \in \{\ell'-1, \ell'-2, \ldots, 0\}$. We split the sequence $f_{\ell'} = \left(f_j^{\ell'}\right)_{j\in\mathbb{Z}}$ into $2^{\ell'-\ell''}$ subsequences

$$f_{\ell',\nu} = \left(f^{\ell'}_{2^{\ell'-\ell''}j+\nu}\right)_{j\in\mathbb{Z}} \qquad \text{for } 0 \le \nu \le 2^{\ell'-\ell''}-1. \tag{A.2}$$

Summation over the indices $2^{\ell'-\ell''}j+\nu$ (ν fixed) yields

$$\sum_{j,k\in\mathbb{Z}} f^{\ell'}_{2^{\ell'-\ell''}j+\nu}\, g_k^{\ell}\, \gamma^{\ell'',\ell',\ell}_{i,2^{\ell'-\ell''}j+\nu,k} = \sum_{j,k\in\mathbb{Z}} f^{\ell'}_{2^{\ell'-\ell''}j+\nu}\, g_k^{\ell}\, \gamma^{\ell'',\ell',\ell}_{i-j,\nu,k} = \sum_{j\in\mathbb{Z}} f^{\ell'}_{2^{\ell'-\ell''}j+\nu}\, G^{\ell'',\ell',\ell}_{i-j,\nu}$$

by the same arguments as before. The right-hand side is the discrete convolution $f_{\ell',\nu} * G_\nu^{\ell'',\ell',\ell}$, where $G_\nu^{\ell'',\ell',\ell}$ is the sequence $\left(G_{i,\nu}^{\ell'',\ell',\ell}\right)_{i\in\mathbb{Z}}$. Summation over ν produces the final result:

$$\omega_{\ell''} := \left(\omega_i^{\ell''}\right)_{i\in\mathbb{Z}} = \sum_{\nu=0}^{2^{\ell'-\ell''}} f_{\ell',\nu} * G_\nu^{\ell'',\ell',\ell}. \tag{A.3}$$

The necessary coefficients $G_{i,\nu}^{\ell'',\ell',\ell}$ are recursively computable via

$$G_{i,\nu}^{\ell'',\ell',\ell} = \tfrac{1}{\sqrt{2}}\left(G_{2i,\nu}^{\ell''+1,\ell',\ell} + G_{2i+1,\nu}^{\ell''+1,\ell',\ell}\right) \qquad \text{for } 0 \le \nu \le 2^{\ell'-\ell''-1}-1, \tag{A.4a}$$

$$G_{i,\nu}^{\ell'',\ell',\ell} = \tfrac{1}{\sqrt{2}}\left(G_{2i-1,\nu-2^{\ell'-\ell''-1}}^{\ell''+1,\ell',\ell} + G_{2i,\nu-2^{\ell'-\ell''-1}}^{\ell''+1,\ell',\ell}\right) \quad \text{for } 2^{\ell'-\ell''-1} \le \nu \le 2^{\ell'-\ell''}-1. \tag{A.4b}$$

For the first step $\ell'' = \ell'-1$, the right-hand side in (A.4a,b) can be expressed by the known coefficients $\Gamma_i^{\ell',\ell}$ (see (A.1)).

A.2 Cost of the Second Version of Step 2b

Step 2a has to be reinterpreted. In the case of the second version of Step 2b the product $\omega_{\ell'} := f_{\ell'} * \Gamma_{\ell',\ell}$ is needed in only $L_{\ell'}$ components ($L_{\ell'} \le N_c(\omega_{\ell'})$ replaces L_c from §6.6). According to Lemma 6.3 and Remark 7.2c, the cost is[14]

$$\sum_{\ell'=0}^{\ell-1} \mathcal{O}\left(\begin{array}{c} \max\left(L_{\ell'}, \min\left(N_c(f_{\ell'}), 2^{\ell'-\ell}N_c(g_\ell)\right)\right) \\ \cdot \log\left(\min\left(L_{\ell'}, N_c(f_{\ell'}), 2^{\ell'-\ell}N_c(g_\ell)\right)\right) \end{array}\right)$$
$$\le \mathcal{O}\left(\sum_{\ell'=0}^{\ell-1} \max\left(L_{\ell'}, N_c(f_{\ell'})\right) \log\left(\min\left(L_{\ell'}, N_c(f_{\ell'})\right)\right)\right).$$

[14] We have replaced $\min\left(N_c(f_{\ell'}), 2^{\ell-\ell'}N_c(g_\ell)\right)$ by the upper bound $N_c(f_{\ell'})$ because of shorter notation.

Before we perform (A.3), i.e., $\omega_{\ell''} := \sum_{\nu=0}^{2^{\ell'-\ell''}} f_{\ell',\nu} * G_\nu^{\ell'',\ell',\ell}$, we have to compute $G_\nu^{\ell'',\ell',\ell}$ for all $0 \le \ell'' \le \ell' \le \ell$ by means of (A.4a,b). Since the range of ν doubles when $\ell'' \to \ell'' - 1$, the cost for computing the G-quantities is $\ell' \cdot \left(2^{\ell'-\ell} N_c(g_\ell)\right)$ for all $0 \le \ell'' \le \ell' - 1$. Summing over all $1 \le \ell' \le \ell$ yields

$$\mathcal{O}\left(\ell N_c(g_\ell)\right).$$

Next, we consider the discrete product $f_{\ell',\nu} * G_\nu^{\ell'',\ell',\ell}$. Note that $N_c(f_{\ell',\nu}) = 2^{\ell''-\ell'} N_c(f_{\ell'})$ and $N_c(G_\nu^{\ell'',\ell',\ell}) = \mathcal{O}\left(2^{\ell''-\ell} N_c(g_\ell)\right)$ (cf. Remark 7.2c). According to Remark 6.2, the convolution cost is $\mathcal{O}\left(\max\left(2^{\ell''-\ell'} N_c(f_{\ell'}), 2^{\ell''-\ell} N_c(g_\ell)\right) \log\left(\min\left(2^{\ell''-\ell'} N_c(f_{\ell'}), 2^{\ell''-\ell} N_c(g_\ell)\right)\right)\right)$. Since for fixed levels ℓ', ℓ'' there are $2^{\ell'-\ell''}$ different values of ν, the convolutions for all ν require $\mathcal{O}\left(\max\left(N_c(f_{\ell'}), 2^{\ell'-\ell} N_c(g_\ell)\right) \log\left(\min\left(N_c(f_{\ell'}), 2^{\ell'-\ell} N_c(g_\ell)\right)\right)\right)$ operations[15]. Summation over $0 \le \ell'' \le \ell'$ introduces an addition factor $(\ell' + 1)$. Further summation over $0 \le \ell' \le \ell$ leads to[15]

$$\mathcal{O}\left(\max\left(\sum_{\ell'=0}^{\ell} (\ell'+1)\, N_c(f_{\ell'}), (\ell+1)\, N_c(g_\ell)\right) \log\left(\min\left(N_c(f_{\ell'}), N_c(g_\ell)\right)\right)\right).$$

B Second Variant in Case B

In this variant, we need to compute the coefficients $G_{\nu,j}^{\ell'',\ell',\ell}$ for few ν and all $j \in \mathbb{Z}$. In Step 1 we start with $\ell'' = \ell$ and proceed with $\ell'' = \ell+1, \ell+2, \ldots, \ell'-1$ in Step 2.

B.1 Step 1: Start

The first part is the computation of $\Gamma_i^{\ell'',\ell} = G_{i,0}^{\ell'',\ell'',\ell} = G_{0,-i}^{\ell'',\ell'',\ell}$ ($\ell'' = \ell, \ell+1, \ldots, 1$, cf. (4.11)). Since these quantities are already known from Step 1b in §5.1.1 (there the index ℓ'' is called ℓ'), this part does not require additional cost.

Note that different from Case A, the first loop involves the index $\ell'' = \ell, \ell+1, \ldots, 1$, while $\ell' = \ell'' - 1, \ldots, 0$ is the second loop. Together, we cover all situations $\ell' < \ell'' \le \ell$ for a fixed ℓ.

B.2 Step 2

The coefficients $G_{i,j}^{\ell'',\ell'',\ell} = \Gamma_{i-j}^{\ell'',\ell}$ are known from Step 1. Now we want to compute $G_{i,j}^{\ell'',\ell',\ell}$ for coarser levels $\ell' = \ell'' - 1, \ldots, 0$. We can write any index $i \in \mathbb{Z}$ as $i' 2^{\ell''-\ell'} + \nu$ with $\nu \in \{0, 1, \ldots, 2^{\ell''-\ell'} - 1\}$. Since

$$G_{i,j}^{\ell'',\ell',\ell} = G_{i'2^{\ell''-\ell'}+\nu,j}^{\ell'',\ell',\ell} = G_{\nu,j-i'}^{\ell'',\ell',\ell},$$

[15] Note that the argument of $\log(\min(\ldots))$ contains the maximum over the summation index.

the data $G_{i,j}^{\ell'',\ell',\ell}$ can be organised by $2^{\ell''-\ell'}$ sequences[16] $\check{G}_{\nu}^{\ell'',\ell',\ell} := (G_{\nu,-j}^{\ell'',\ell',\ell})_{j\in\mathbb{Z}}$, $\nu = 0, \ldots, 2^{\ell''-\ell'} - 1$. For their computation use the following formulae for $\ell' = \ell'' - 1, \ldots, 0$:

$$G_{\nu,j}^{\ell'',\ell',\ell} = \frac{1}{\sqrt{2}} \left(G_{\nu,2j}^{\ell'',\ell'+1,\ell} + G_{\nu,2j+1}^{\ell'',\ell'+1,\ell} \right) \qquad \text{for } 0 \le \nu \le 2^{\ell''-\ell'-1} - 1, \tag{B.1a}$$

$$G_{\nu+2^{\ell''-\ell'-1},j}^{\ell'',\ell',\ell} = \frac{1}{\sqrt{2}} \left(G_{\nu,2j-1}^{\ell'',\ell'+1,\ell} + G_{\nu,2j}^{\ell'',\ell'+1,\ell} \right) \qquad \text{for } 0 \le \nu \le 2^{\ell''-\ell'-1} - 1. \tag{B.1b}$$

Together, the coefficients $G_{\nu,j}^{\ell'',\ell',\ell}$ are defined for all $0 \le \nu \le 2^{\ell''-\ell'} - 1$ and all $j \in \mathbb{Z}$.

For a proof of (B.1a,b) use $G_{\nu,j}^{\ell'',\ell',\ell} = \frac{1}{\sqrt{2}} \left(G_{\nu,2j}^{\ell'',\ell'+1,\ell} + G_{\nu,2j+1}^{\ell'',\ell'+1,\ell} \right)$ (cf. (4.10b)) and

$$\begin{aligned}
G_{\nu+2^{\ell''-\ell'-1},2j}^{\ell'',\ell'+1,\ell} &= \sum_{k\in\mathbb{Z}} g_k^{\ell} \gamma_{\nu+2^{\ell''-\ell'-1},2j,k}^{\ell'',\ell'+1,\ell} \underset{(4.6)}{=} \sum_{k\in\mathbb{Z}} g_k^{\ell} \gamma_{k-\nu-2^{\ell''-\ell'-1}+(2j)2^{\ell''-\ell'-1}}^{\ell'',\ell'+1,\ell} \\
&= \sum_{k\in\mathbb{Z}} g_k^{\ell} \gamma_{k-\nu+(2j-1)2^{\ell''-(\ell'+1)}}^{\ell'',\ell'+1,\ell} = G_{\nu,2j-1}^{\ell'',\ell'+1,\ell}, \\
G_{\nu+2^{\ell''-\ell'-1},2j+1}^{\ell'',\ell'+1,\ell} &= \sum_{k\in\mathbb{Z}} g_k^{\ell} \gamma_{k-\nu-2^{\ell''-\ell'-1}+(2j+1)2^{\ell''-\ell'-1}}^{\ell'',\ell'+1,\ell} = \sum_{k\in\mathbb{Z}} g_k^{\ell} \gamma_{k-\nu+(2j)2^{\ell''-\ell'-1}}^{\ell'',\ell'+1,\ell} = G_{\nu,2j}^{\ell'',\ell'+1,\ell}.
\end{aligned}$$

B.3 Step 3

The projection $P_{\ell''}(f_{\ell'} * g_{\ell})$ has the coefficients $\omega_i^{\ell''}$ which are split into $2^{\ell''-\ell'}$ subsequences (indexed again by i with fixed ν):

$$\omega_{2^{\ell''-\ell'}i+\nu}^{\ell''} = \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} \, g_k^{\ell} \, \gamma_{2^{\ell''-\ell'}i+\nu,j,k}^{\ell'',\ell',\ell} \qquad \text{for } 0 \le \nu \le 2^{\ell''-\ell'} - 1 \text{ and } i \in \mathbb{Z}.$$

Since $\sum_{k\in\mathbb{Z}} g_k^{\ell} \, \gamma_{2^{\ell''-\ell'}i+\nu,j,k}^{\ell'',\ell',\ell} = \sum_{k\in\mathbb{Z}} g_k^{\ell} \, \gamma_{k-(2^{\ell''-\ell'}i+\nu)2^{\ell-\ell''}+j2^{\ell-\ell'}}^{\ell'',\ell',\ell} = \sum_{k\in\mathbb{Z}} g_k^{\ell} \, \gamma_{k-\nu 2^{\ell-\ell''}+(j-i)2^{\ell-\ell'}}^{\ell'',\ell',\ell} = \sum_{k\in\mathbb{Z}} g_k^{\ell} \, \gamma_{\nu,j-i,k}^{\ell'',\ell',\ell} = G_{\nu,j-i}^{\ell'',\ell',\ell}$, we have

$$\omega_{2^{\ell''-\ell'}i+\nu}^{\ell''} = \sum_{j\in\mathbb{Z}} f_j^{\ell'} \, G_{\nu,j-i}^{\ell'',\ell',\ell} \qquad \text{for } 0 \le \nu \le 2^{\ell''-\ell'} - 1 \text{ and } i \in \mathbb{Z}.$$

Hence, the subsequence $\omega_{\ell'',\nu} := \left(\omega_{2^{\ell''-\ell'}i+\nu}^{\ell''} \right)_{i\in\mathbb{Z}}$ is the discrete convolution $f_{\ell'} * \check{G}_{\nu}^{\ell'',\ell',\ell}$ with $\check{G}_{\nu}^{\ell'',\ell',\ell} := \left(G_{\nu,-j}^{\ell'',\ell',\ell} \right)_{j\in\mathbb{Z}}$:

$$\omega_{\ell'',\nu} := f_{\ell'} * \check{G}_{\nu}^{\ell'',\ell',\ell} \qquad \left(0 \le \nu \le 2^{\ell''-\ell'} - 1 \right). \tag{B.2}$$

Together, the sequences $\omega_{\ell'',\nu}$, $0 \le \nu \le 2^{\ell''-\ell'} - 1$, represent all coefficients of $\omega_{\ell''} = P_{\ell''}(f_{\ell'} * g_{\ell})$.

[16] The definition uses $-j$ in $(G_{\nu,-j}^{\ell'',\ell',\ell})_{j\in\mathbb{Z}}$ in order to obtain a standard discrete convolution in (B.2).

B.4 Combining the Computations for all $\ell' < \ell'' \le \ell$

Again, we use the already computed $\Gamma_i^{\ell''}$ from (5.6) instead of $\Gamma_i^{\ell'',\ell}$, to treat all levels $\ell = \ell'', \ell''+1, \ldots, L$ at once. The further algorithm is

algorithm	*explanations:*
for $\ell'' := 1$ **to** L **do**	
for $\ell' := \ell'' - 1$ **downto** 0 **do**	
begin $G_{0,j}^{\ell'',\ell''} := \Gamma_{-j}^{\ell''}$;	starting value,
for $\nu := 0$ **to** $2^{\ell''-\ell'-1}$ **do**	
begin $G_{\nu,j}^{\ell'',\ell'} = \frac{1}{\sqrt{2}}\left(G_{\nu,2j}^{\ell'',\ell'+1} + G_{\nu,2j+1}^{\ell'',\ell'+1}\right)$;	recursion (B.1a),
$G_{\nu+2^{\ell''-\ell'-1},j}^{\ell'',\ell',\ell} = \frac{1}{\sqrt{2}}\left(G_{\nu,2j-1}^{\ell'',\ell'+1,\ell} + G_{\nu,2j}^{\ell'',\ell'+1,\ell}\right)$	recursion (B.1b),
end;	
for $\nu := 0$ **to** $2^{\ell''-\ell'}$ **do** $\omega_{\ell'',\nu} := f_{\ell'} * \check{G}_\nu^{\ell'',\ell'}$	see (B.2).
end;	

The resulting sequences $\omega_{\ell''}$ contain all contributions $P_{\ell''}(f_{\ell'} * g_\ell)$ with $\ell' < \ell'' \le \ell$.

B.5 Corresponding Cost

As already mentioned, Step 1 does not require additional cost.

In Step 2 we have to perform (B.1a,b) for $\ell' = \ell'' - 1, , \ldots, 0$ and $0 \le \nu \le 2^{\ell''-\ell'} - 1$. With $\ell' \to \ell' - 1$ the length of $\check{G}_\nu^{\ell'',\ell',\ell} = \left(G_{\nu,-j}^{\ell'',\ell',\ell}\right)_{j\in\mathbb{Z}}$ is halved, but since the range $0 \le \nu \le 2^{\ell''-\ell'} - 1$ doubles, we have $\sum_{\nu=0}^{2^{\ell''-\ell'}-1} N_c(\check{G}_\nu^{\ell'',\ell',\ell}) \le \sum_{\nu=0}^{2^{\ell''-\ell'}-1} \mathcal{O}\left(2^{\ell'-\ell} N_c(g_\ell)\right) = \mathcal{O}\left(2^{\ell''-\ell} N_c(g_\ell)\right)$ (cf. Lemma 7.1 and Remark 7.2c). Summation over $1 \le \ell'' \le \ell$ yields the bound $N_c(g_\ell)$ for the data size. The number of operations needed for their computation is proportional:

$$\mathcal{O}\left(N_c(g_\ell)\right). \tag{B.3}$$

In Step 3 the discrete convolutions $\omega_{\ell'',\nu} := f_{\ell'} * \check{G}_\nu^{\ell'',\ell',\ell}$ are to be performed for all $0 \le \nu \le 2^{\ell''-\ell'} - 1$. We need $L_{\ell''}$ components of $\omega_{\ell''}$. Since $\omega_{\ell''}$ is split into $2^{\ell''-\ell'}$ subsequences $\omega_{\ell'',\nu}$, in the average we have to compute $2^{\ell'-\ell''} L_{\ell''}$ components of $\omega_{\ell'',\nu}$. The corresponding cost is

$$\begin{aligned}
&\sum_{\ell''=1}^{\ell} \sum_{\ell'=0}^{\ell''-1} 2^{\ell''-\ell'} \cdot \mathcal{O}\left(\begin{array}{c} \max\left(2^{\ell'-\ell''} L_{\ell''}, \min\left(N_c(f_{\ell'}), 2^{\ell'-\ell} N_c(g_\ell)\right)\right) \\ \cdot \log\left(\min\left(2^{\ell'-\ell''} L_{\ell''}, N_c(f_{\ell'}), 2^{\ell'-\ell} N_c(g_\ell)\right)\right) \end{array} \right) \\
&\le \sum_{\ell'=0}^{\ell-1} \sum_{\ell''=\ell'+1}^{\ell} \mathcal{O}\left(\max\left(L_{\ell''}, 2^{\ell''-\ell} N_c(g_\ell)\right) \log\left(\min\left(L_{\ell''}, N_c(f_{\ell'}), 2^{\ell''-\ell} N_c(g_\ell)\right)\right)\right) \\
&= \sum_{\ell'=0}^{\ell-1} \mathcal{O}\left(\max\left(\sum_{\ell''=\ell'+1}^{\ell} L_{\ell''}, N_c(g_\ell)\right) \log\left(\min\left(\max_{\ell''} L_{\ell''}, N_c(f_{\ell'}), N_c(g_\ell)\right)\right)\right) \\
&\le \mathcal{O}\left(\max\left(\sum_{\ell''=1}^{\ell} \ell'' L_{\ell''}, \ell N_c(g_\ell)\right) \log\left(N_c(g_\ell)\right)\right)
\end{aligned} \tag{B.4}$$

(cf. Lemma 6.3 and Remark 7.2c).

C Second Variant in Case C

C.1 Computation of δ

C.1.1 Definitions

We define new $\check{\gamma}$-coefficients[17]

$$\check{\gamma}_{i,j,k}^{\ell,\ell',\ell} := \int \Phi_j^{\ell'}(y)\Phi_k^{\ell}(ih_\ell - y)\mathrm{d}y \qquad (0 \le \ell' \le \ell,\ i,j,k \in \mathbb{Z})$$

($\Phi_i^\ell(x)$ replaced by the Dirac function at ih_ℓ). Simple substitutions yield

$$\begin{aligned}\check{\gamma}_{i,j,k}^{\ell,\ell',\ell} &= \int \Phi_0^{\ell'}(y - jh_{\ell'})\Phi_0^{\ell}((i-k)\,h_\ell - y)\mathrm{d}y \\ &= \int \Phi_0^{\ell'}(y)\Phi_0^{\ell}((i-k)\,h_\ell - (y + jh_{\ell'}))\mathrm{d}y = \int \Phi_0^{\ell'}(y)\Phi_0^{\ell}\left((i-k-j2^{\ell-\ell'})h_\ell - y\right)\mathrm{d}y \\ &= \int \Phi_0^{\ell'}(y)\Phi_{k-i+j2^{\ell-\ell'}}^{\ell}(-y)\mathrm{d}y =: \check{\gamma}_{k-i+j2^{\ell-\ell'}}^{\ell',\ell},\end{aligned}$$

defining simplified coefficients $\check{\gamma}_k^{\ell',\ell}$ with one subindex. The values can be given explicitly:

$$\check{\gamma}_k^{\ell',\ell} = \begin{cases} 2^{-(\ell-\ell')/2} & \text{for } -1 \ge k \ge -2^{\ell-\ell'}, \\ 0 & \text{otherwise.}\end{cases}$$

Instead one may use the recursion

$$\check{\gamma}_k^{\ell,\ell} = \begin{cases} 1 & \text{for } k = -1, \\ 0 & \text{otherwise,}\end{cases} \tag{C.1a}$$

$$\check{\gamma}_k^{\ell',\ell} = \frac{1}{\sqrt{2}}\left(\check{\gamma}_k^{\ell'+1,\ell} + \check{\gamma}_{k+2^{\ell-\ell'-1}}^{\ell'+1,\ell}\right) \qquad \text{for } \ell' < \ell, \tag{C.1b}$$

which follows from

$$\check{\gamma}_{k-i+j2^{\ell'-\ell}}^{\ell',\ell} = \check{\gamma}_{i,j,k}^{\ell,\ell',\ell} = \frac{1}{\sqrt{2}}\left(\check{\gamma}_{i,2j,k}^{\ell,\ell'+1,\ell} + \check{\gamma}_{i,2j+1,k}^{\ell,\ell'+1,\ell}\right) = \frac{1}{\sqrt{2}}\left(\check{\gamma}_{k-i+j2^{\ell-\ell'}}^{\ell'+1,\ell} + \check{\gamma}_{k-i+j2^{\ell-\ell'}+2^{\ell-\ell'-1}}^{\ell'+1,\ell}\right),$$

so that for $i = j = 0$ the identity $\check{\gamma}_k^{\ell',\ell} = \frac{1}{\sqrt{2}}\left(\check{\gamma}_k^{\ell'+1,\ell} + \check{\gamma}_{k+2^{\ell-\ell'-1}}^{\ell'+1,\ell}\right)$ is derived.

The δ-values are

$$\begin{aligned}\delta_i = (f_{\ell'} * g_\ell)(ih_\ell) &= \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell} \int \Phi_j^{\ell'}(y)\Phi_k^{\ell}(ih_\ell - y)\mathrm{d}y = \sum_{j,k\in\mathbb{Z}} f_j^{\ell'} g_k^{\ell}\, \check{\gamma}_{i,j,k}^{\ell,\ell',\ell} \\ &= \sum_{j\in\mathbb{Z}} f_j^{\ell'}\, \check{G}_{i,j}^{\ell,\ell',\ell},\end{aligned}$$

where we have introduced

$$\check{G}_{i,j}^{\ell,\ell',\ell} := \sum_{k\in\mathbb{Z}} g_k^{\ell}\, \check{\gamma}_{i,j,k}^{\ell,\ell',\ell}.$$

[17] As for $\gamma_{i,j,k}^{\ell'',\ell',\ell}$ we write $\check{\gamma}_{i,j,k}^{\ell,\ell',\ell}$ with three superindices. However, the first one is not really needed because of $\ell'' = \ell$.

C.1.2 Step 1: $\ell' = \ell$

For $\ell' = \ell$ we have to compute $\check{G}^{\ell,\ell,\ell}_{i,j} := \sum_{k\in\mathbb{Z}} g^\ell_k \, \check{\gamma}^{\ell,\ell,\ell}_{i,j,k} = \sum_{k\in\mathbb{Z}} g^\ell_k \, \check{\gamma}^{\ell,\ell}_{k-i+j}$. Using (C.1a) we obtain

$$\check{G}^{\ell,\ell,\ell}_{i,j} = \check{G}^{\ell,\ell,\ell}_{i-j,0} = \check{G}^{\ell,\ell,\ell}_{0,j-i} = g^\ell_{i-j-1} \qquad (i,j\in\mathbb{Z})\,.$$

C.1.3 Step 2: $\ell' = \ell-1,\ldots,0$

For $\ell' < \ell$, the recursion

$$\check{G}^{\ell,\ell',\ell}_{i,j} = \frac{1}{\sqrt{2}}\left(\check{G}^{\ell,\ell'+1,\ell}_{i,2j} + \check{G}^{\ell,\ell'+1,\ell}_{i,2j+1}\right)$$

holds again (cf. (4.10b)). Note that for $i = \mu 2^{\ell-\ell'} + \nu$ $(0 \le \nu \le 2^{\ell-\ell'}-1)$

$$\check{G}^{\ell,\ell',\ell}_{i,j} = \sum_{k\in\mathbb{Z}} g^\ell_k \, \check{\gamma}^{\ell,\ell',\ell}_{i,j,k} = \sum_{k\in\mathbb{Z}} g^\ell_k \, \check{\gamma}^{\ell',\ell}_{k-i+j2^{\ell-\ell'}} = \check{G}^{\ell,\ell',\ell}_{i-j2^{\ell-\ell'},0} = \check{G}^{\ell,\ell',\ell}_{\nu+(\mu-j)2^{\ell-\ell'},0} = \check{G}^{\ell,\ell',\ell}_{\nu,j-\mu}\,,$$

so that the recursion becomes

$$\check{G}^{\ell,\ell',\ell}_{\nu,j} = \tfrac{1}{\sqrt{2}}\left(\check{G}^{\ell,\ell'+1,\ell}_{\nu,2j} + \check{G}^{\ell,\ell'+1,\ell}_{\nu,2j+1}\right) \qquad \text{for } 0 \le \nu \le 2^{\ell-\ell'-1}-1, \tag{C.2a}$$

$$\check{G}^{\ell,\ell',\ell}_{\nu,j} = \tfrac{1}{\sqrt{2}}\left(\check{G}^{\ell,\ell'+1,\ell}_{\nu-2^{\ell-\ell'-1},2j-1} + \check{G}^{\ell,\ell'+1,\ell}_{\nu-2^{\ell-\ell'-1},2j}\right) \qquad \text{for } 2^{\ell-\ell'-1} \le \nu \le 2^{\ell-\ell'}-1. \tag{C.2b}$$

Write the index i as

$$i = \mu 2^{\ell-\ell'} + \nu \qquad \text{with } 0 \le \nu \le 2^{\ell-\ell'}-1,\ \mu\in\mathbb{Z}.$$

The computation of

$$\delta_{\mu 2^{\ell-\ell'}+\nu} = \sum_{j,k\in\mathbb{Z}} f^{\ell'}_j \, g^\ell_k \, \check{\gamma}^{\ell,\ell',\ell}_{\mu 2^{\ell-\ell'}+\nu,j,k} = \sum_{j\in\mathbb{Z}} f^{\ell'}_j \, \check{G}^{\ell,\ell',\ell}_{\mu 2^{\ell-\ell'}+\nu,j} = \sum_{j\in\mathbb{Z}} f^{\ell'}_j \, \check{G}^{\ell,\ell',\ell}_{\nu,j-\mu} \tag{C.3}$$

needs a discrete convolution to obtain the subsequence $\delta^{(\nu)} := \left(\delta_{\mu 2^{\ell-\ell'}+\nu}\right)_{\mu\in\mathbb{Z}}$. Performing the convolutions for all $0 \le \nu \le 2^{\ell-\ell'}-1$, we have completed the whole sequence $\delta = (\delta_\nu)_{\nu\in\mathbb{Z}}$.

We remark that we need the values δ_ν and only $\delta_{\nu+1}$ for those $\nu\in\mathbb{Z}$ for which I^ℓ_ν contains one of the intervals $I^{\ell''}_i$ with $\ell'' > \ell$ and $i \in \mathcal{I}^\omega_{\ell''}$ (cf. (3.1)).

C.1.4 Combining the Computations for all $\ell' \le \ell < \ell''$

	explanations:
for $\ell := 0$ **to** $L-1$ **do**	
begin for $\ell' := \ell$ **downto** 0 **do**	
begin if $\ell' = \ell$ **then begin** $\check{G}^{\ell,\ell,\ell}_{0,-i} = g^{\ell}_{i-1}$; $\delta := 0$ **end**	starting values,
else for $\nu := 0$ **to** $2^{\ell-\ell'-1}-1$ **do**	case of $\ell' < \ell$,
begin $\check{G}^{\ell,\ell',\ell}_{\nu,j} = (\check{G}^{\ell,\ell'+1,\ell}_{\nu,2j} + \check{G}^{\ell,\ell'+1,\ell}_{\nu,2j+1})/\sqrt{2}$;	see (C.2a),
$\check{G}^{\ell,\ell',\ell}_{\nu+2^{\ell-\ell'-1},j} = (\check{G}^{\ell,\ell'+1,\ell}_{\nu,2j-1} + \check{G}^{\ell,\ell'+1,\ell}_{\nu,2j})/\sqrt{2}$	see (C.2b),
end;	
for $\nu := 0$ **to** $2^{\ell-\ell'}-1$ **do**	
begin compute $\delta^{(\nu)}$ from (C.3); add $\delta^{(\nu)}$ to δ	see (C.3),
end end;	
for $\ell'' := \ell+1$ **to** L **do**	
for all $i = \nu 2^{\ell''-\ell} + j$ **with** $0 \le j \le 2^{\ell''-\ell}-1$ **do**	
$\omega^{\ell''}_i = \sqrt{h_{\ell''}}\left(\delta_\nu + \frac{i+\frac{1}{2}}{2^{\ell''-\ell}}(\delta_{\nu+1}-\delta_\nu)\right)$	see (5.12).
end;	

C.2 Corresponding Cost

First, we have to compute the $\check{G}$-coefficients. For $\ell' = \ell$, the $\check{G}^{\ell,\ell,\ell}_{0,j}$ are given by g_ℓ proving $N_c(\check{G}^{\ell,\ell,\ell}_{0,*}) = N_c(g_\ell)$, where we denote the sequence by $\check{G}^{\ell,\ell',\ell}_{\nu,*} := \left(\check{G}^{\ell,\ell',\ell}_{\nu,j}\right)_{j\in\mathbb{Z}}$ (we remark that only $\ell'' = \ell$ appears). The computations start with the recursion (C.2a,b). The length is $N_c(\check{G}^{\ell,\ell',\ell}_{\nu,*}) \le 2^{\ell'-\ell}N_c(g_\ell)+1$ (cf. Lemma 7.1). The number of different ν-values is $2^{\ell-\ell'}$. This leads to $\le \mathcal{O}(N_c(g_\ell))$ operations (cf. Remark 7.2c). Summation over all $0 \le \ell' \le \ell$ yields the upper bound

$$(\ell+1)\,N_c(g_\ell) \tag{C.4}$$

The subsequence $\delta^{(\nu)} := \left(\delta_{\mu 2^{\ell-\ell'}+\nu}\right)_{\mu\in\mathbb{Z}}$ requires one convolution in (C.3) and is to be evaluated for $L_{\ell,\nu}$ indices. The sum of the latter numbers is

$$\sum_{\nu=0}^{2^{\ell-\ell'}-1} L_{\ell,\nu} =: L_{(\ell)} \le \sum_{\ell''=\ell+1}^{L^\omega} N_{cc}(\omega_{\ell+1})/2+2.$$

Proof. $N_c(\omega_{\ell''})$ is the number of indices i where $\omega^{\ell''}_i$ is to be evaluated. The related intervals are $I^{\ell''}_i$, $i \in \mathcal{I}^\omega_{\ell''}$. The intervals $I^{\ell''}_i$ for all $i \in \mathcal{I}^\omega_{\ell''}$ and all $\ell < \ell'' \le L^\omega$ intersect at most $\sum_{\ell''=\ell+1}^{L^\omega} N_{cc}(\omega_{\ell+1})/2+1$ intervals of $\mathcal{M}_\ell$. The number of grid points is by one larger. ■

The convolution cost for all $0 \le \ell' \le \ell$ and $0 \le \nu \le 2^{\ell-\ell'}$ is

$$\sum_{\ell'=0}^{\ell}\sum_{\nu=0}^{2^{\ell-\ell'}-1}\mathcal{O}\left(\begin{array}{l}\max\left(L_{\ell,\nu},\min\left(N_c(f_{\ell'}),N_c(\check{G}^{\ell,\ell',\ell}_{\nu,*})\right)\right)\\ \cdot\log\left(\min\left(L_{\ell,\nu},N_c(f_{\ell'}),N_c(\check{G}^{\ell,\ell',\ell}_{\nu,*})\right)\right)\end{array}\right)$$

$$=\sum_{\ell'=0}^{\ell}\mathcal{O}\left(\begin{array}{l}\max\left(L_{(\ell)},\min\left(2^{\ell-\ell'}N_c(f_{\ell'}),N_c(g_\ell)\right)\right)\\ \cdot\log\left(\min\left(L_{(\ell)},N_c(f_{\ell'}),2^{\ell'-\ell}N_c(g_\ell)\right)\right)\end{array}\right)$$

$$=\mathcal{O}\left((\ell+1)\max\left(L_{(\ell)},N_c(g_\ell)\right)\log\left(\min\left(L_{(\ell)},\max_{\ell'=0}^{\ell}N_c(f_{\ell'}),N_c(g_\ell)\right)\right)\right). \tag{C.5}$$