A MULTIRESOLUTION SCHEME FEATURING FULLY ADAPTIVE BLOCKS FOR SIMULATING REACTIVE FLOWS

A PREPRINT

Brandon Gusto
Department of Scientific Computing
Florida State University
Tallahassee, FL 32306
bgusto@fsu.edu

Tomasz Plewa
Department of Scientific Computing
Florida State University
Tallahassee, FL 32306
tplewa@fsu.edu

June 13, 2019

ABSTRACT

We present a generalization of Harten's original multiresolution scheme for simulating reactive flows on logically rectangular block-structured adaptive mesh refinement (AMR) grids in one and two dimensions. The scheme addresses a major shortcoming of tree-based AMR codes, which is the creation of blocks with a low filling factor; that is, many cells in such a block are resolved beyond the desired error tolerance, necessitating excessive computational resources. To overcome this issue, we introduce a multiresolution representation of the solution, not only to adapt the grid but also to adaptively compute fluxes and sources on each block. The scheme recycles regularity information obtained by the multiresolution grid adaptation in order to select flux and source calculations which may be accurately replaced by interpolation from the multiresolution basis. A block which employs this scheme is denoted as a fully adaptive block (FAB). The error introduced by this approximation is shown to be of the same order as the local truncation error of the reconstruction scheme. Thus the rate of convergence of the underlying spatial reconstruction scheme is preserved. Additionally with respect to parallel applications, the multiresolution transform and computation of fluxes and sources on FABs is asynchronous, requiring only one synchronization step which is equivalent to the filling of ghost cells for each block. The efficiency of the scheme is demonstrated using several one and two-dimensional problems.

***K*eywords** Multiresolution · Adaptive Mesh Refinement · Conservation Laws

1 Introduction

Fluid flows are often characterized by disparate spatial and temporal length scales. Certain features such as turbulence or shocks necessitate significantly higher resolution than smooth regions of the flow. In reacting flows, combustion fronts often act in highly localized regions, and on very small time scales. Naturally, intense effort has gone into the development of methods which employ a multi-scale or adaptive strategy to accurately simulate such flows without over-resolving smooth areas of the domain.

The most popular strategy to accurately capture regions of interest in fluid simulations is to introduce a non-uniform spatial grid, with higher resolution in regions of interest. Methods which introduce a hierarchy of nested grid resolutions are generally described as adaptive mesh refinement (AMR) methods. AMR methods rely on estimates of the local truncation error (LTE) of the numerical scheme to determine regions where refinement is necessary for solution accuracy. Several strategies to approximate the LTE are used (list methods).

Regarding the implementation of AMR methods on large parallel computers, certain engineering realities have neccessitated the reduction in granularity of the adaptive refinement. To use a single computational cell as the unit for refinement (i.e. cell-based refinement) introduces a number of costly compromises. Firstly, such an adapted grid

requires the reconstruction method of choice to utilize nonuniform stencils, requiring increased computational resources. More significantly, the cell-based refinement requires extremely costly data traversal. Traversing tree space requires an average $\mathcal{O}(n^d)$ (confirm this?) operations. Thus most AMR codes make use of some type of block-based approach. Tree-based block-structured codes, where each block consists of a fixed number of cells, are a very popular choice. These types of approaches are implemented in a number of AMR libraries including Paramesh (cite), p4est (cite, and others. This approach allows for simple mesh management procedures, and scales well for very large number of processors in parallel.

Alternate approaches to dynamic grid adaptation based on wavelet theory have become popular in recent years. The first such effort was introduced in a seminal paper by Harten [**?**], where a multiresolution representation of the discrete solution on a uniform grid was used for adaptively computing the divergence of the flux within a finite volume framework. Rather than adapt the grid by truncating the wavelet basis, the idea was to accelerate the computation of high-order essentially non-oscillatory (ENO) schemes using the multiresolution information. Fluxes in smooth regions were interpolated from fluxes obtained at interfaces corresponding to coarser grid levels. The original scheme was applied solely to hyperbolic conservation laws, but was then expanded by Bihari et. al. to two-dimensional simulations in (citation), followed by the inclusion of viscous terms in (citation), and then tp source terms in the context of reactive flows in (citation). These works retained the original flavor of Harten's scheme, which was to represent the discrete solution on a uniform grid, but uses a multiresolution representation of the solution to identify regions where flux (and source term) computations may be avoided. The multiresolution transform is obtained by using average-interpolating wavelets as basis functions.

Although Harten's original scheme was intended to be an alternative to spatially non-uniform grid adaptation, a series of papers have since reintroduced this concept of non-uniform grids within the MR framework. Thus the AMR approach has been redeveloped but with the refinement criterion defined by the MR representation rather than with the traditional metrics mentioned previously. The first fully adaptive scheme was presented by Cohen et. al. to study hyperbolic conservation laws in two dimensions in (cite).

2 Governing Equations and Finite Volume Discretization

In the present work we are interested in numerically solving conservation laws of the form

$$\begin{cases} u_t + f(u)_x = s(u) \\ u(x,0) = u_0(x), \end{cases} \tag{1}$$

where u represents a conserved quantity, $f(u)$ is the flux function, and $s(u)$ is a source term. For the sake of presentation, we let the scalar equation (1) stand in for the more complicated systems of conversation laws which will be the focus of our applications. In the finite volume formulation, the solution $u(x,t)$ is approximated by volume averages defined within each cell $I_i = \left[x_i - \frac{h}{2}, x_i + \frac{h}{2}\right]$ in the computational domain. The cell width h is determined by the number of cells, N, and the size of the domain $\Omega \in [a,b]$. The cell averages are given by

$$u_i(t) = \frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}} u(\xi,t)d\xi, \tag{2}$$

where for convenience we use $i \pm 1/2$ in the subscripts to indicate the left and right interfaces of the target cell (i.e. $x_{i+1/2} = x_i + \frac{h}{2}$). The governing equations are cast into the semi-discrete conservative form,

$$\frac{du_i(t)}{dt} = -\frac{1}{h}\left(\hat{f}_{i+1/2} - \hat{f}_{i-1/2}\right), \tag{3}$$

where the numerical flux is evaluated as

$$\hat{f}_{i+1/2} = \hat{f}(u^-_{i+1/2}, u^+_{i+1/2}). \tag{4}$$

In the present work, the reconstruction method of choice is a fifth-order weighted essentially non-oscillatory (WENO) scheme.

2.1 WENO

2.2 Time Integration

Once the system of ordinary differential equations (3) is established, the objective is to integrate them forward in time. In our implementation we use a second-order explicit Runge-Kutta scheme to advance. This is summarized by

$$\boldsymbol{u}^{n+1} = \boldsymbol{E}\boldsymbol{u}^n, \tag{5}$$

where $\boldsymbol{E}$ represents the nonlinear evolution operator, and codifies the temporal and spatial discretizations.

order	γ_1	γ_2	γ_3	γ_4
3	1/8	0	0	0
5	-22/128	3/128	0	0
7	0	0	0	0
9	0	0	0	0

Table 1: For the derivation of these coefficients, the reader is referred to Appendix (ref).

3 Harten's Multiresolution Scheme on a Uniform Grid

We present here a review of Harten's multiresolution method for adaptive flux computations within the original context of uniform, one-dimensional grids. This is necessary to illustrate the extension of the scheme to block-structured AMR grids. The multiresolution approach is based on a hierarchy of nested discretizations on the domain of interest, Ω. The hierarchy consists of grids at a coarsest level $l = 1$ to a finest grid $l = L$, where any two adjacent levels differ by a refinement factor of two. The grids are defined by

$$\boldsymbol{\mathcal{G}}_l = \left\{x_i^l\right\}_{i=0}^{N_l}, \quad x_i^l = i \cdot h_l, \;\; h_l = 2^{L-l} \cdot h_L, \;\; N_l = N_L/2^{L-l}, \tag{6}$$

where h_l and N_l denote the cell width and number of cells, respectively, on level l. We denote the finest grid by $\boldsymbol{\mathcal{G}}_L$. denote the index space of cells on each level of the hierarchy by $\boldsymbol{\mathcal{I}}^l = (1, \ldots, N^l)$.

Encoding

Given a vector of discrete cell averages at the finest resolution, $\boldsymbol{u}^L$, the multiresolution representation is obtained by the following operations:

Split: The cells at grid level l are split into even and odd sets, $\boldsymbol{\mathcal{I}}_{even}^l$ and $\boldsymbol{\mathcal{I}}_{odd}^l$, based on their indices.

Project: The cells at level $l+1$ are projected by means of volume averaging, onto the coarser grid level l. The projection is defined by a linear operator which performs the mapping $\boldsymbol{P}_{l+1}^l : \boldsymbol{u}^{l+1} \mapsto \boldsymbol{u}^l$.

Predict: A stencil of cell averages $i \in \boldsymbol{\mathcal{I}}_{odd}^l$ are predicted by an average-interpolating polynomial based on cells at level $l-1$. Thus, the prediction operator performs the mapping $\boldsymbol{P}_{l-1}^l : \boldsymbol{u}^{l-1} \mapsto \boldsymbol{u}^l$.

The projection preserves averages at the finer level, by design, and in the context of uniform grids is the simple averaging of parent cells,

$$u_i^l = \left(\boldsymbol{P}_{l+1}^l \boldsymbol{u}^{l+1}\right)_i = \frac{1}{2}(u_{2i-1}^{l+1} + u_{2i}^{l+1}), \quad \forall i \in \boldsymbol{\mathcal{I}}^l. \tag{7}$$

The prediction operator is defined by a unique average-interpolating polynomial to predict, based on a stencil of cell-averages at grid level l, the odd cell averages at the finer level level $l+1$. The prediction mapping is performed using the following centered interpolants,

$$\tilde{u}_i^l = \left(\boldsymbol{P}_{l-1}^l \boldsymbol{u}^{l-1}\right)_i = \begin{cases} u_{i/2}^{l-1} - \sum_{p=1}^s \gamma_p \left(u_{i/2-p}^{l-1} - u_{i/2+p}^{l-1}\right), & \forall i \in \boldsymbol{\mathcal{I}}_{even}^l \\ u_{(i+1)/2}^{l-1} + \sum_{p=1}^s \gamma_p \left(u_{(i+1)/2-p}^{l-1} - u_{(i+1)/2+p}^{l-1}\right), & \forall i \in \boldsymbol{\mathcal{I}}_{odd}^l \end{cases} \tag{8}$$

The coefficients γ_i are supplied in Table (3). The order of accuracy of each interpolant is $r = 2s + 1$. Once the prediction is made, the difference information on level l is obtained by computing the detail coefficient as

$$d_i^l = u_{2i}^{l+1} - \tilde{u}_{2i}^{l+1}, \quad \forall i \in \boldsymbol{\mathcal{I}}^l. \tag{9}$$

for each cell. The encoding procedure is complete when the detail coefficients have been computed on all levels $l = 0, \ldots, L-1$. The entire procedure can be succinctly written in terms of a matrix-vector operation to yield the multiresolution representation of the data, u_M^L, as

$$\boldsymbol{u}_M^L = \boldsymbol{M}\boldsymbol{u}^L = \left(\boldsymbol{d}^{L-1}, \boldsymbol{d}^{L-2}, \ldots, \boldsymbol{d}^0, \boldsymbol{u}^0\right)^T. \tag{10}$$

Here the multiresolution operator $\boldsymbol{M}$ contains the prediction operation and subsequent detail coefficient calculation for each cell on each level of the hierarchy.

order	β_1	β_2	β_3	β_4
3	9/16	-1/16	0	0
5	0	0	0	0
7	0	0	0	0
9	0	0	0	0

Table 2:

Truncation

Once the difference information has been computed, compression of the fine-grid cell averages $\boldsymbol{u}^L$ is obtained by truncating coefficients whose magnitude is below a prescribed level-dependent threshold, ε_l. In [**?**] the following threshold is proposed

$$\varepsilon_l = \varepsilon / 2^{L-l}. \tag{11}$$

The detail coefficients are truncated according to

$$\tilde{d}_i^l = \begin{cases} d_i^l, \text{ if } |d_i^l| > \varepsilon_l \\ 0, \text{ if } |d_i^l| \leq \varepsilon_l, \end{cases} \tag{12}$$

producing the following approximate representation,

$$\tilde{\boldsymbol{u}}_M^L = T_\varepsilon(\boldsymbol{u}_M^L) = \left(\tilde{\boldsymbol{d}}^{L-1}, \tilde{\boldsymbol{d}}^{L-2}, \ldots, \tilde{\boldsymbol{d}}^0, \boldsymbol{u}^0\right)^T. \tag{13}$$

Decoding

The fine-grid solution is reconstructed by computing from levels $l = 1, \ldots, L-1$ the following

$$u_{2i}^{l+1} = d_i^l + \tilde{u}_{2i}^{l+1} \tag{14}$$

$$u_{2i-1}^{l+1} = 2u_i^l - u_{2i}^{l+1}, \tag{15}$$

for each cell on the level. If the detail coefficients are truncated below a nonzero ε, the discrete representation becomes

$$\boldsymbol{u}_\varepsilon^L \approx \boldsymbol{M}^{-1} \tilde{\boldsymbol{u}}_M^L. \tag{16}$$

Adaptive Calculation of Fluxes and Sources

Once the detail coefficients have been obtained, the MR scheme proceeds by setting a threshold ε and truncating coefficients which have an absolute value below the threshold. Lastly, the inverse transform then starts from grid $l = L$ and at each interface either computes fluxes using the fine-grid scheme, or interpolates them using the MR basis. The fluxes are interpolated by

$$\tilde{f}_{2i+1}^{l-1} \approx \sum_{p=1}^{s+1} \beta_p \left(\hat{f}_{i-p+1}^l + \hat{f}_{i+p}^l\right), \tag{17}$$

where the interpolants are of degree $2s+1$. The coefficients for various degrees of polynomial interpolants are shown in Table (ref). The process repeats until all fluxes are either computed or interpolated on the fine grid $l = 0$.

Error Analysis

4 Asynchronous Fully Adaptive Block Scheme

Block-Structured Mesh Adaptation

We define a mask $\boldsymbol{\mathcal{M}}(\varepsilon)$.

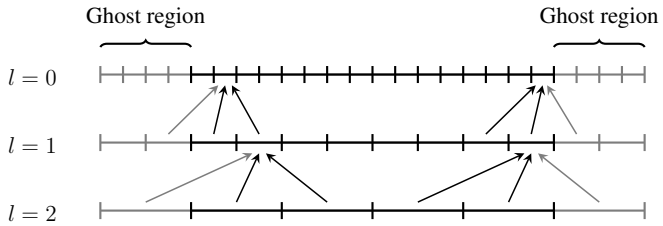

Figure 1: A block of consisting of $N^0 = 16$ cells is shown. Four ghost cells are included on each end of the block, allowing the multiresolution decomposition to descend two levels (to grid level $l = 2$). Interpolation stencils for the computation of detail coefficients at levels $l = 1, l = 2$ are shown, indicating the need for ghost cells.

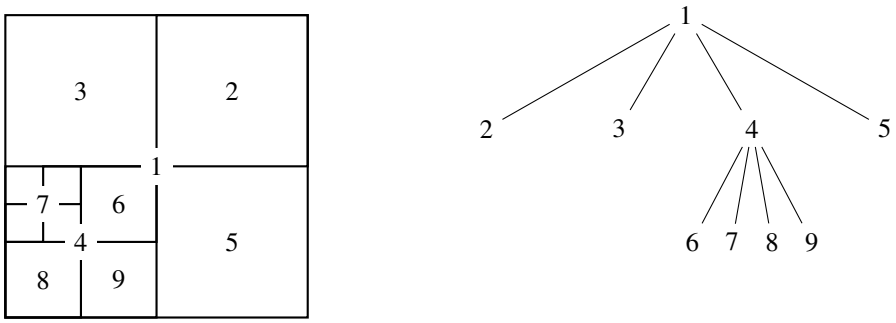

Figure 2:

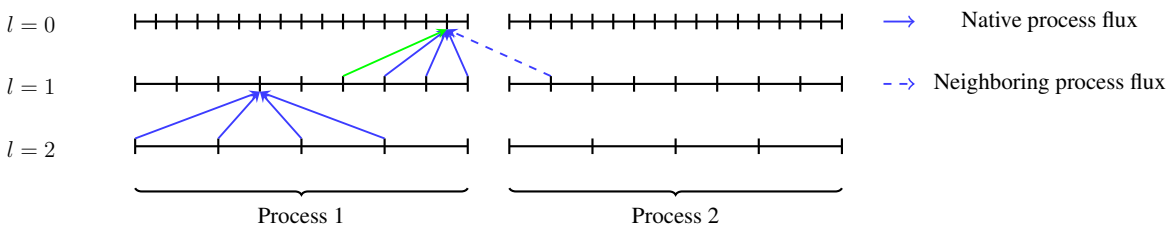

Figure 3: Two examples of flux interpolation on a hierarchy of grids on Process 1: one procedure requires flux data from the adjacent process, the other does not.

Buffer Region

Load Balancing

5 Numerical Results

Convergence Analysis for Smooth Advection Problem

	$\varepsilon = 0.0$				$\varepsilon = 10^{-12}$			
grid cells	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
16								
32								
64								
128								
256								
	$\varepsilon = 10^{-6}$				$\varepsilon = 10^{-4}$			
grid cells	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
16								
32								
64								
128								
256								

Interacting Blast Waves

Nuclear Burning

Mach Reflection

Using the inviscid flow assumption, the dynamics of compressible fluids are modeled using the reactive Euler equations add domain notation

$$u_t + f(u)_x + g(u)_y = s(u), \tag{18}$$

where $u = (\rho, \rho u, \rho v, \rho w, E)^T$ is the state vector, the flux vectors are given by

$$f = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{pmatrix}, \quad g = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{pmatrix}, \tag{19}$$

and $s(u)$ represents sources. The total energy per unit volume is given by

$$E = \rho \left(\frac{1}{2}\mathbf{V}^2 + e \right),$$

where e is the internal energy and the kinetic energy contribution is

$$\frac{1}{2}\mathbf{V}^2 = \frac{1}{2}\mathbf{V} \cdot \mathbf{V} = \frac{1}{2}\left(u^2 + v^2\right).$$

The system of nonlinear equations is closed by an equation of state which is in general not derived from that of an ideal gas.

6 Acknowledgements

A Multiresolution Analysis

A multiresolution analysis (MRA) of the Lebesgue space $L^2(\mathbb{R})$ defines a sequence of nested approximation spaces. These spaces satisfy certain self-similarity properties in both space and scale. An MRA defines a sequence of closed subspaces $\{\mathcal{V}_j : j \in \mathbb{Z}\}$ such that

$$\mathcal{V}_0 \subset \mathcal{V}_1 \subset \mathcal{V}_2 \subset \cdots \subset L^2.$$

The complement of $\mathcal{V}_j \in \mathcal{V}_{j+1}$ is defined by $\mathcal{W}_j$, known as the detail space. This relation is defined by a direct summation as

$$\mathcal{V}_{j+1} = \mathcal{W}_j \oplus \mathcal{V}_j.$$

Considering successively finer approximation spaces yields for any arbitrary level J,

$$\mathcal{V}_J = \mathcal{V}_0 \oplus \mathcal{W}_0 \oplus \mathcal{W}_1 \oplus \cdots \oplus \mathcal{W}_{J-1}.$$

Thus fine-scale information on any arbitrary level J is represented by the coarsest scale plus a series of differences at higher levels. Interested readers can refer to (cite) for more details on the construction of the bi-orthogonal multiresolution analysis used.

B Derivation of Prediction Operator in One-Dimension

We are interested in obtaining the difference between approximation spaces at varying levels of resolution. We are given cell-averaged values as input data to our wavelet transform. This data is fed to the scheme at some arbitrary maximum resolution level J, and the wavelet transform produces details coefficients at each lower level until the coarsest level, $j = 0$, is reached. The coefficients in this case are interchangeable with the cell-averages and are denoted by c_k^j, where the level of resolution is denoted by j, and the spatial index is denoted by k. We consider an interpolating polynomial $p(x)$ such that

$$c_{k-1}^j = \int_{x_{k-1}^j}^{x_k^j} p(x)dx \tag{20}$$

$$c_k^j = \int_{x_k^j}^{x_{k+1}^j} p(x)dx \tag{21}$$

$$c_{k+1}^j = \int_{x_{k+1}^j}^{x_{k+2}^j} p(x)dx. \tag{22}$$

The polynomial $p(x)$ should then predict the finer cell-averages of cell c_k^j as

$$\hat{c}_{2k}^{j+1} = 2\int_{x_k^j}^{x_{k+1/2}^j} p(x)dx \tag{23}$$

$$\hat{c}_{2k+1}^{j+1} = 2\int_{x_{k+1/2}^j}^{x_{k+1}^j} p(x)dx \tag{24}$$

At present, it may not be clear how to implement such a scheme on a computer. However this interpolation procedure can be cast in a more suitable form by introducing another polynomial, the integral of $p(x)$:

$$P(x) = \int_0^x p(y)dy. \tag{25}$$

Now the problem is to interpolate the following data

$$0 = P(x_{k-1}^j) \tag{26}$$

$$c_{k-1}^j = P(x_k^j) \tag{27}$$

$$c_{k-1}^j + c_k^j = P(x_{k+1}^j) \tag{28}$$

$$c_{k-1}^j + c_k^j + c_{k+1}^j = P(x_{k+2}^j). \tag{29}$$

This can easily be done using Lagrange polynomials. Then the predictions are given in terms of $P(x)$ by

$$\hat{c}_{2k}^{j+1} = 2\left(P(x_{k+1/2}^j) - P(x_k^j)\right) \tag{30}$$

$$\hat{c}_{2k+1}^{j+1} = 2\left(P(x_{k+1}^j) - P(x_{k+1/2}^j)\right). \tag{31}$$

This interpolating polynomial is cast in the Lagrange form,

$$P(x) = \sum_{i=0}^{n} y_i l_i(x), \tag{32}$$

where y_i are the functional data, and $l_i(x)$ are the Lagrange polynomials. For $n = 3$ these are given by

$$l_0(x) = \frac{x - x_1}{x_0 - x_1}\frac{x - x_2}{x_0 - x_2}\frac{x - x_3}{x_0 - x_3} \tag{33}$$

$$l_1(x) = \frac{x - x_0}{x_1 - x_0}\frac{x - x_2}{x_1 - x_2}\frac{x - x_3}{x_1 - x_3} \tag{34}$$

$$l_2(x) = \frac{x - x_0}{x_2 - x_0}\frac{x - x_1}{x_2 - x_1}\frac{x - x_3}{x_2 - x_3} \tag{35}$$

$$l_3(x) = \frac{x - x_0}{x_3 - x_0}\frac{x - x_1}{x_3 - x_1}\frac{x - x_2}{x_3 - x_2}, \tag{36}$$

and the final interpolating polynomial is

$$P(x) = (0)l_0(x) + (c_{k-1}^j)l_1(x) + (c_{k-1}^j + c_k^j)l_2(x) + (c_{k-1}^j + c_k^j + c_{k+1}^j)l_3(x). \tag{37}$$

Several evaluations are necessary in order to obtain the predictions. Using intervals of equal length, these values are

$$P(x_k^j) = c_{k-1}^j \tag{38}$$

$$P(x_{k+1/2}^j) = \frac{17}{16}c_{k-1}^j + \frac{1}{2}c_k^j - \frac{1}{16}c_{k+1}^j \tag{39}$$

$$P(x_{k+1}^j) = c_{k-1}^j + c_k^j. \tag{40}$$

Then the predictions of the cell-averages at the higher level of resolution are finally given by

$$\hat{c}_{2k}^{j+1} = c_k^j + \frac{1}{8}\left(c_{k-1}^j - c_{k+1}^j\right) \tag{41}$$

$$\hat{c}_{2k+1}^{j+1} = c_k^j - \frac{1}{8}\left(c_{k-1}^j - c_{k+1}^j\right). \tag{42}$$

This procedure could easily be extended to non-uniformly spaced intervals, giving different weights. Note that only the odd indices are counted because in the multiresolution scheme the data is initially split into even and odd signals. All data at level j are just considered to be a copy of the even-index data at level $j + 1$, whereas the odd-indexed data at level $j + 1$ is what is predicted by even-indexed data at level $j + 1$. Also important are the interpolants at the ends of the domain. Given below are the left and right predictions, respectively:

$$\hat{c}_{2k+1}^{j+1} = \frac{5}{8}c_k^j + \frac{1}{2}c_{k+1}^j - \frac{1}{8}c_{k+2}^j \tag{43}$$

$$\hat{c}_{2k+1}^{j+1} = \frac{1}{8}c_{k-2}^j - \frac{1}{2}c_{k-1}^j + \frac{11}{8}c_k^j. \tag{44}$$