Causal–Path Local Time–Stepping in the Discontinuous Galerkin Method for Maxwell's equations

L. D. Angulo[a], J. Alvarez[b], F. L. Teixeira[c], M. F. Pantoja[a], S. G. Garcia[a]

[a] *Department of Electromagnetism and Matter Physics, University of Granada, Granada 18071, Spain (e-mail: lmdiazangulo@ugr.es)*
[b] *Cassidian, EADS-CASA, Getafe 28906, Spain (e-mail: jesus@ieee.org)*
[c] *ElectroScience Laboratory and the Department of Electrical and Computer Engineering, The Ohio State University, Columbus, Ohio, 43212 USA (e-mail: teixeira@ece.osu.edu)*

Abstract

We introduce a novel local time-stepping technique for marching-in-time algorithms. The technique is denoted as Causal-Path Local Time-Stepping (CPLTS) and it is applied for two time integration techniques: fourth order low–storage explicit Runge–Kutta (LSERK4) and second order Leapfrog (LF2). The CPLTS method is applied to evolve Maxwell's curl equations using a Discontinuous Galerkin (DG) scheme for the spatial discretization. Numerical results for LF2 and LSERK4 are compared with analytical solutions and the Montseny's LF2 technique. The results show that the CPLTS technique improves the dispersive and dissipative properties of LF2-LTS scheme.

Keywords: Causal Path, Local Time–Stepping, LTS, Discontinuous Galerkin Methods, Maxwell's equations, DGTD

1. Introduction

Many time–stepping algorithms have been proposed in order to improve the performance of Discontinuous Galerkin (DG) based schemes by increasing the maximum time step while preserving stability. There are usually two kinds of strategies used for this purpose: to use implicit schemes [1, 2] or, to use a explicit local time–stepping (LTS) technique [3, 4, 5, 2, 6, 7, 8, 9]. An advantage of LTS schemes versus implicit strategies is that the former can be used recursively and easily paralellized. Moreover, highly disparate mesh element sizes can lead to ill-conditioning problems in implicit schemes which are obviated by explicit schemes. Additionally, time integration algorithms may have other constraints on the time–step arising from accuracy considerations and other inherent time scales such as in dispersive media [10] or when hybridized with network/lumped elements models [11], LTS techniques can also contribute to mitigate these problems in a simple and straightforward way.

When a second order convergent spatial discretization is used, the most commonly used time integration method is the second–order leapfrog (LF2) algorithm. Several authors [4, 5] use a LF2-LTS scheme proposed by Montseny [6] consisting of using the last known values of the fields on the larger time stepped region each time that the smaller one needs a field value. Piperno [2] adopts a similar approach based on a Verlet

Preprint submitted to Journal of Computational Physics *September 2, 2013*

Published in Journal of Computational Physics,
vol. 256, pp. 678-695, 2014

scheme. Alvarez [3, 12, 13, 14] contributed with a novel approach to perform LTS in LF2 schemes whereby an interpolation between the fields is used in an interface between the larger and smaller time–stepped regions. A rigorous demonstration of the stability and dispersive properties of these schemes is still an open problem.

Diaz and Grote [7, 8] implemented a rigorous study on the stability and dispersion of LF-LTS high–order schemes applied to the second–order wave equation by means of an eigenvalue analysis. They found that the LTS introduces numerical dispersion and can produce instabilities if the global time step is not slightly reduced with respect to a classic implementation. The authors also found that the global stability could be improved by enlarging the smaller time–stepped region.

For higher order methods, explicit Runge-Kutta (RK) algorithms [15, 16, 17, 18, 19, 20] seem to be preferred with respect to LF schemes [21]. Despite of their popularity, there are less works in the literature related to RK-LTS than to LF-LTS. In [22] we find an RK-LTS scheme in which all elements are integrated with the least restrictive time step and then the interface with the region where the polluted solution has not been able to affect is used as a boundary condition for the regions which require substepping. Specifically for Maxwell's equations, RK-LTS algorithms usually rely on interpolations at the interfaces using previously computed solutions [15] or arbitrary high-order derivatives (ADER) schemes [9, 23, 24].

In this paper we present a novel LTS technique that can be applied to a large variety of time integration algorithms. It does not need interpolation between computed solutions and nor directly uses any previously known values. Numerical results showing comparisons with analytical solutions for applications on a second–order Leap-Frog (LF2) and on a fourth–order Low Storage Explicit Runge–Kutta scheme (LSERK4) are shown to demonstrate the advantages of the proposed LTS technique.

2. Discontinuous Galerkin Semidiscretization

Maxwell's curl equations for source–less homogeneous media can be written as

$$\begin{aligned} \vec{\nabla} \times \vec{E} &= -\mu \partial_t \vec{H} \\ \vec{\nabla} \times \vec{H} &= \varepsilon \partial_t \vec{E} \end{aligned} \tag{1}$$

For simplicity, in our discussion we will assume that ε and μ do not vary in the computational domain, and use a system of units where $\varepsilon = \mu = 1$.

We tessellate the computational domain with $k = 1, \dots, K$ non–overlapping tetrahedrons. In each of those, we apply the Discontinuous Galerkin's formalism [15, 17, 18] to obtain

$$\begin{aligned} \mathcal{M}_k \partial_t \mathbf{E}_k(t) + \mathcal{S}_k \mathbf{H}_k(t) - \sum_f \mathcal{F}_{kf} \mathbf{H}^*_{kf}(t) &= 0 \\ \mathcal{M}_k \partial_t \mathbf{H}_k(t) + \mathcal{S}_k \mathbf{E}_k(t) - \sum_f \mathcal{F}_{kf} \mathbf{E}^*_{kf}(t) &= 0 \end{aligned} \tag{2}$$

With $\mathcal{M}$ being the mass matrix, $\mathcal{S}$ the spatial semidiscretization of the curl operator and $\mathcal{F}_f$ the lift operator for face f. $\mathbf{E}$ and $\mathbf{H}$ are column vectors containing all the degrees of freedom for the electric and magnetic field respectively. $\mathbf{E}^*$ and $\mathbf{H}^*$ are the numerical fluxes.

We define a state vector $\mathbf{q}_k = [\mathbf{E}_k\ \mathbf{H}_k]^T$ containing all the N_k degrees of freedom of element k. With this definition, we can rewrite system (2) as a single equation that governs the time evolution of the system,

$$\partial_t \mathbf{q}_k(t) = -(\mathcal{M}_k^q)^{-1}\left(\mathcal{S}_k^q \mathbf{q}_k(t) - \sum_f \mathcal{F}_{kf}^q \left(\mathcal{E}_{kf}\mathbf{q}_k(t) - \mathcal{E}_{kf_+}\mathbf{q}_{kf_+}(t)\right)\right) \tag{3}$$

The DG method gives us some freedom in the selection of the operators $\mathcal{E}_{kf}$ and $\mathcal{E}_{kf_+}$ as long as it respects the properties of consistency, continuity, and monotonicity needed for the numerical flux [25]. If this operator is block diagonal with all its components being 1/2, we will say that the semi-discrete scheme is using a centered flux and therefore is numerically non-dissipative [26, 3]. On the other hand, if these operators are non-block diagonal we will say that the flux is being penalized and therefore the semi-discrete scheme is numerically dissipative. We will mostly focus on a particular case of penalized flux: the upwind flux [17, 27], coming from the solution of the Riemann problem.

When using penalized fluxes some dissipation is introduced and more operations are needed to compute the flux terms. However, introducing such penalization is known to improve numerical dispersion and suppress spurious modes [6, 17, 3, 14, 15, 28]. Altough the contribution to dissipation coming from penalized fluxes may be negligible at well-resolved frequencies, it may become important at higher frequencies for sufficiently long integration times; therefore, care should be exercised by the user depending on the application.

To simplify the discussion further we will change the basis of the vector space using an invertible operator $\mathcal{P}_k$ on equation (3) that diagonalizes only the locally applied operators,

$$\mathcal{W}_k = -\mathcal{P}_k^{-1}(\mathcal{M}_k^q)^{-1}(\mathcal{S}_k^q - \sum_f \mathcal{F}_{kf}^q \mathcal{E}_{kf})\mathcal{P}_k \tag{4}$$

We can also define the eigenmodes as

$$\mathbf{p}_k = \mathcal{P}_k^{-1}\mathbf{q}_k \tag{5}$$

and the external operators as

$$\mathcal{V}_{kf} = -\mathcal{P}_k^{-1}(\mathcal{M}^q)_k^{-1}\mathcal{F}_{kf}^q \mathcal{E}_{kf_+}\mathcal{P}_k \tag{6}$$

This change of basis let us write equation (3) in the following compact form

$$\partial_t \mathbf{p}_k(t) = \mathcal{W}_k \mathbf{p}_k(t) + \sum_f \mathcal{V}_{kf}\mathbf{p}_{kf_+}(t) \tag{7}$$

3. Time integration

In the following discussion, we will focus on two time integration methods that are also the most popular choices in conjunction with DG semidiscretizations.

3.1. Second-order Leap-Frog (LF2)

The second-order leap-frog method [29] is applied by alternately evolving the $\mathbf{E}^n$ and $\mathbf{H}^{n+1/2}$ fields, arbitrarily defined at times t_n and $t_n + \Delta t/2$ respectively. This implies that we do not have a fully defined state vector in the sense of eq. (3) for a given time t. To obtain the future values from a present state the following algorithm is applied

$$\begin{aligned} \mathbf{E}^{n+1} &= \mathbf{E}^n + \Delta t\ \mathcal{L}_h\left(\mathbf{H}^{n+1/2}, \mathbf{E}^n\right) \\ \mathbf{H}^{n+3/2} &= \mathbf{H}^{n+1/2} + \Delta t\ \mathcal{L}_h\left(\mathbf{E}^n, \mathbf{H}^{n+1/2}\right) \end{aligned} \tag{8}$$

With $\mathcal{L}_h$ being a function representing the result of applying the spatial semi–discretization. When centered fluxes are used, the operator $\mathcal{L}_h$ only uses $\mathbf{H}^{n+1/2}$ or $\mathbf{E}^n$ as arguments. This implies that the scheme is reversible in time and will preserve energy as long as the time step used is below a maximum value Δt_k set by a CFL-like condition [1, 2, 29].

3.2. Low-Storage Explicit Runge–Kutta (LSERK4)

The second method that we will use in our discussion is the five-stage fourth-order Explicit Runge-Kutta method (LSERK4) [15, 20, 30]. This method states that for a given vector representing the state of the system, i.e. $\mathbf{p}_k(t) = \mathbf{p}_k^n$ we can find an approximate solution state $\mathbf{p}_k(t + \Delta t) = \mathbf{p}_k^{n+1}$ applying the following algorithm

$$\begin{aligned} \mathbf{p}_k^{(0)} &= \mathbf{p}_k^n, \\ \mathbf{r}^{(i)} &= a_i \mathbf{r}^{(i-1)} + \Delta t \left(\mathcal{W}_k \mathbf{p}_k^{(i-1)} + \sum_f \mathcal{V}_{kf} \mathbf{p}_{kf_+}^{(i-1)} \right), \\ \mathbf{p}_k^{(i)} &= \mathbf{p}_k^{(i-1)} + b_i \mathbf{r}^{(i)}, \\ \mathbf{p}_k^{(n+1)} &= \mathbf{p}_k^{(5)} \end{aligned} \tag{9}$$

with $i \in [1, ..., 5]$ and the coefficients a_i, b_i and c_i taking the values indicated in Table 1. The LSERK4 scheme is one of the most used methods in high–order Discontinuous Galerkin semi–discretizations, because it introduces low dispersion and dissipation. Contrary to other RK implementations, the low–storage version requires the storage of only two times the number of degrees of freedom in the scheme at the expense of one additional stage. RK methods are constrained by the spectra of the operator $\mathcal{W}_k$, i.e. all the eigenvalues of $\mathcal{W}_k$ must lie inside of the stability region of the RK scheme. Consequently, the time step must be chosen sufficiently small, e.g. for a nodal basis the following inequality must hold [15]

$$\Delta t_k \leq \frac{C}{c_k} \min_i \frac{\Delta r_{ki}}{2} \tag{10}$$

where $\min_i \Delta r_{ki}$ indicates the minimum distance between nodes in element k and c_k is the maximum speed of light in the element k.

Despite its many advantages, LSERK4 has a high computational cost and the numerical dissipation it introduces can be a factor depending on the application.

s	a_s	b_s	c_s
1	0	$\frac{1432997174477}{9575080441755}$	0
2	- $\frac{567301805773}{1357537059087}$	$\frac{5161836677717}{13612068292357}$	$\frac{13612068292357}{9575080441755}$
3	- $\frac{2404267990393}{2016746695238}$	$\frac{1720146321549}{2090206949498}$	$\frac{22526269341429}{6820363962896}$
4	- $\frac{-3550918686646}{2091501179385}$	$\frac{3134564353537}{4481467310338}$	$\frac{2006345519317}{3224310063776}$
5	- $\frac{1275806237668}{842570457699}$	$\frac{2277821191437}{14882151754819}$	$\frac{28032321613138}{2924317926251}$

Table 1: Coefficients for the low-storage five-stages fourth-order Explicit Runge–Kutta method (LSERK4)

4. The Causal–Path LTS technique

In this section we introduce the Causal–Path technique as a novel way of performing LTS in different time integration techniques. We require two basic properties for the time integration technique:

1. It has to provide a fully defined state $\mathbf{q}_k(t)$ for each element.
2. The next state $\mathbf{q}_k(t + \Delta t)$ can be explicitly computed from a neighbourhood of elements.

As a first step we will organize the elements in different groups, called tiers, according to their time steps denoted as Δt^m. An element k will belong to a tier $m = [0, \ldots, N_m - 1]$ if its maximum time step Δt_k is such that

$$\Delta t^m \leq \Delta t_k < \Delta t^{m+1} \tag{11}$$

In order to compute the next time step, we need to use the field values at local and neighbor elements, $\mathbf{p}_k^{(m,i-1)}$ and $\mathbf{p}_{kf_+}^{(m,i-1)}$. If there is no connection with other elements belonging to a lower tier, we can evolve all the elements in m using their Δt^m. However, in the border between a tier m and $m+1$ we can not apply the direct algorithm because the value $\mathbf{p}_{k^m f_+}^{(m,i-1)} = \mathbf{p}_{k^{m+1}}^{(m,i-1)}$ has not been computed.

The strategy that we propose is to compute the values $\mathbf{p}_{k^{m+1}}^{(m,i-1)}$ using $\Delta t^{m-1} = h_{i-1}\Delta t^m$ as time step wherever they are necessary. If to do that, we need additional neighbour values that have not been computed, we recursively apply this idea until a known value is found. Thus, starting from $m = 0$ we can compute all the stages needed to evolve it before starting with the tier $m = 1$ and so on. Finally, the values $\mathbf{p}_{k^{m+1}}^{(m,i-1)}$ are casted aside and the upper tier uses the original values from the lower tier.

To compute the next time step values in each of the N_m tiers we may need to compute N_s stages in all the elements of tier m. We will also need to compute intermediate stages between the stages in the $m + 1$ tier. So, in order to avoid a possible interleaving with other higher tiers, we impose that the $(N_s - 1)$–depth neighbourhood of a tier m is only composed of elements belonging to tier $m+1$ or $m-1$. This additional condition for the tier assortment is illustrated in Figure 1.

The implementation of this algorithm may seem difficult at a first glance; however, the recursive nature of the algorithm allows us to make use of recursive calls to the function used to evolve the system. Every time the function is called, we pass the information about the tier in which this is being computed and the time step that has to be applied. So starting from a call to evolve the N_m tier for a given time step Δt, the function will recursively call itself on each of the stages of the algorithm passing $N_m - 1$ and $h_i \Delta t$ as arguments and evolving its corresponding tier elements. This technique also requires that the degrees of freedom in the region being interfaced are saved in the higher tier. Note that no interpolation of field values is necessary and only past field values generated by the discretization itself are utilized. This is a desirable property because schemes that perform interpolation are not reversible in time and thus dissipative. However, the fact that we are casting away values used in the intermediate states of lower tiers makes our scheme also non–reversible. Altough the idea of creating a discrete domain-of-dependence is not new, our technique differs from the one presented in [22] because we are not interpolating at the interface with a non polluted solution to obtain the values that would be needed by the lower tier. Rather than that, the lower tier evolves using the values obtained from applying the same integration technique with its maximum time step. In the next sections we describe two examples of the CPLTS technique, applied to the LF2 and LSERK4 algorithms, together with illustrations to clarify the concepts.

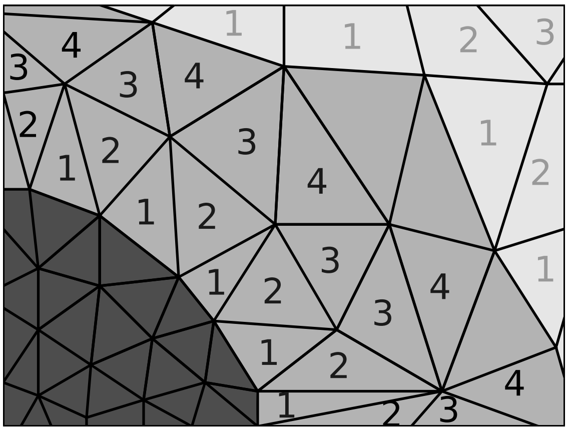


Figure 1: This figure illustrates the concept of 4–depth neigbourhood of two different regions. The darker colors indicate elements belonging to a lower temporal tier and thus having a smaller time step.

4.1. LF2-CPLTS

Since the LF2 performs iterations using a single stage we can create any distribution of N_s intermediate stages in the higher tiers to fit the evaluations needed by the smaller tiers. The time–steps of the intermediate stages would then be $h_i \Delta t^{m+1,i} = \Delta t^m$, with $h_i > 0$ and the restriction $\sum_i^{N_s} h_i = 1$. The choice of $h_i = 1/N_s$ would be the most favourable in terms of computational cost. Figure 2 and Algorithm 1 show an schematic view of this scheme applied to the case $h_1 = h_2 = 1/2$. Note that this freedom in choosing

h_i is an improvement compared with the Montseny's scheme [6], which is constrained due to the field interleaving of the LF2 scheme by the condition $\Delta t^m = \Delta t^{m-1}(1+2k)$, where k is a positive integer number. This is also an improvement with respect to the Verlet–Piperno's scheme [2] in which $\Delta t^{m+1} = 2\Delta t^m$, and it allows our scheme to adapt to the different transitions as necessary; however, for the sake of simplicity we will not consider these cases here.

On the other hand, we need both values of $\mathbf{E}$ and $\mathbf{H}$ at same time instants in order to find a fully–defined state of the system at any given stage $\mathbf{p}^{(m,i)}$. In other words, we can not apply this LTS technique computing only $\mathbf{E}(t_n)$ and $\mathbf{H}(t_n + \Delta t/2)$ because to compute the intermediate value of a lower tier, let us say $\mathbf{E}^{m-1}(t_n + \Delta t^m/2)$ we would need the values of the magnetic field $\mathbf{H}^m(t_n)$. To overcome this issue we need to apply LF2 twice, doubling the computational costs with respect to the conventional approach.

When we apply this scheme to a non–dissipative semi–discretization (e.g. DG with centered flux) we find that the scheme is unstable showing growing high–frequency numerical modes. The introduction of a penalized flux solves this problem through higher frequency damping [16, 26].

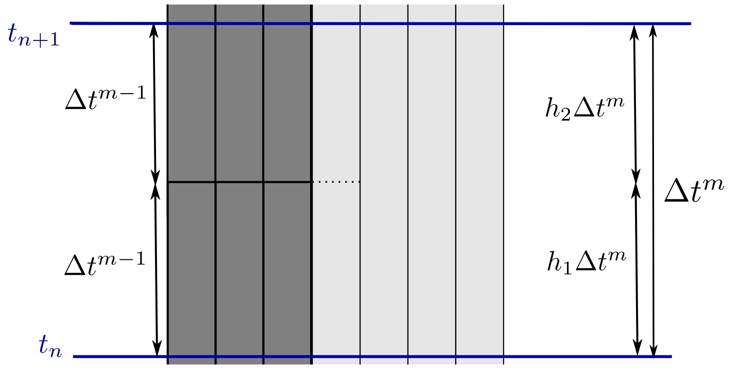


Figure 2: Schematic view of the LF2-CPLTS algorithm for the case $h_1 = h_2 = 1/2$. Vertical lines indicate boundaries between elements. At t_n all the field values are known. With a maximum time step Δt^{m-1}, the darker region needs to apply twice the time integration algorithm to reach t_{n+1}. The LF2-CPLTS approach applies the first integration also in the elements of the neighbourhood of the darker region (marked with a dashed line). With those values at $t_n + \Delta t^{m-1}$ we can apply the algorithm again to reach t_{n+1}. The lighter region t_{n+1} values are obtained using the original values in t_n with a Δt^m time integration.

4.2. LSERK4-CPLTS

When the CPLTS technique is applied to an LSERK4 (Algorithm 2) we note that the stages are not evenly distributed in time. As a result, we apply a variable time step in the lower tiers (Figure 3). The values for Δt^m and Δt^{m-1} are chosen such that equation (10) is always enforced and therefore

$$\max_i(h_i)\Delta t^m = \Delta t^{m-1} \tag{12}$$

with $\max_i(h_i)$ being the maximum stage size (for LSERK4 $\max_i(h_i) = h_4 = c_5 - c_4 = 0.336026 \simeq 1/3$). Whenever we compute intermediate stages in higher tiers we satisfy

Algorithm 1: LF2-CPLTS algorithm with two stages and $h_1 = h_2 = 1/2$. To evolve from $\mathbf{q}(t_n)$ to $\mathbf{q}(t_{n+1})$ this algorithm is called twice using (t_n), $(\Delta t^{N_m-1}/2)$ and $(N_m - 1)$ as input arguments. The superscript RHS refers to Right Hand Side.

```
Data: Inputs: $t$, $\Delta t$, $m$
Result: $\mathbf{q}_k(t)$ is evolved to $\mathbf{q}_k(t + \Delta t)$
if $m = N_m - 1$ then
    $k$ are all the elements in tier $m$.
else
    $k$ are all the elements in tier $m$ and its 1–neighbourhood.
end
$\mathbf{q}_k^{\text{RHS}} \leftarrow \mathcal{L}_h(\mathbf{q}_k)$
if $m > 0$ then
    $k^*$ are the elements in the 1-neighbourhood of $m - 1$.
    $\mathbf{q}_{k^*}^{\text{Saved}} \leftarrow \mathbf{q}_{k^*}$
    Calls this algorithm with $(t + c_s\Delta t)$, $(h_s\Delta t)$ and $(m - 1)$ as inputs.
    $\mathbf{q}_{k^*} \leftarrow \mathbf{q}_{k^*}^{\text{Saved}}$
end
$\mathbf{q}_k \leftarrow \mathbf{q}_k + \mathbf{q}_k^{\text{RHS}}\Delta t$
```

this condition because in higher tiers this condition is less restrictive. However, every time we apply this division, N_s times more computational operations are needed to get a speed-up of about three times in the higher tier region. So, if the largest tier region is not at least 5/3 times larger than the smallest we would not see any appreciable global speed-up.

For this reason it seems preferable to organize the time tiers with $\Delta t^{m-1} = \Delta t^m/N_s$ rather than with the maximum stage size criteria. By doing this, we are computing an stage in the lower tier region with a time-step bigger than is strictly allowed based on a conventional CFL-like criterion for the associated direct algorithm, which could be a source of potential instability. On the other hand, the smaller stages in the lower tier compute the solution using a time-step smaller than the maximum allowed and thus introducing an additional numerical dissipation. We may then wonder if the additional dissipation introduced by the smaller stages offsets the potential for instability introduced by the larger. Note that as long as these effects are mostly kept limited to high frequency components (which are under-resolved anyway) the solution accuracy should not be impacted. In the next sections we perform some tests to assess the practical validity of this approach.

5. Numerical Results

In this section we present comparisons between results using the proposed CPLTS technique, the LF2-LTS technique introduced by Montseny [6], classical implementations of the algorithms, and analytical solutions.

For all cases we use nodal basis of order $P = 2$ and numerical upwind fluxes as described in [15, 17, 18, 31]. This implies that we are using 60 degrees of freedom

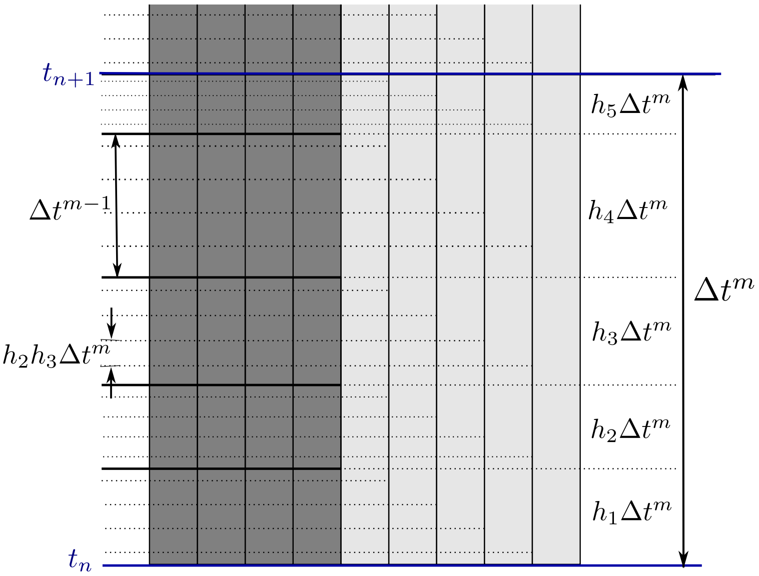


Figure 3: LSERK4-CPLTS sketch. Additional operations are made only in the 4-depth neighbourhood of the smaller tier region (darker). In this sketch the darker region has a maximum time step $\Delta t^{m-1} = h_4\Delta t^m$ corresponding to the largest stage of the LSERK4 scheme. As with the LF2-CPLTS case, the darker region is evolved first to reach $t_n + h_1\Delta t^m$. To do so, the field values marked with a dashed line will be needed for the different stages of the LSERK4 algorithm applied for a time step $h_1\Delta t^m$. Once the darker region has evolved to $t_n + \Delta t^m$, the lighter one computes its first stage from the original values in t_n. The process continues in a similar fashion until all the stages of the ligher region are computed and both regions are in the state t_{n+1}. Note that the darker region never uses a time step bigger than its maximum, Δt^{m-1}

.

per element. The implementation has been performed with an in–house C++ code[1] with OpenMP parallelization[2]. GiD was used to obtain meshes and for pre and post-processing[3]. Simulations for the reflection and resonance problems were performed using a single processor laptop with Intel(R) Core(TM)2 Duo CPU T9400 @ 2.53GHz processor and running Ubuntu 12.04 LTS. The RCS problem were run in a desktop computer with an Intel(R) Core(TM) i7-3960X CPU @ 3.30GHz processor with 12 cores and Ubuntu 10.04 LTS.

5.1. Reflection caused by a non-homogeneous mesh

The first example we present is an study of the numerical reflection caused by differences in the mesh size, a similar type of analysis can be found in [32, 33]. This type of analysis is important for LTS because it quantifies a source of additive noise on the results. Figure 4 shows the meshes used, together with an isometric view of the boundary conditions employed. A plane wave excitation with z-polarization is introduced in one of

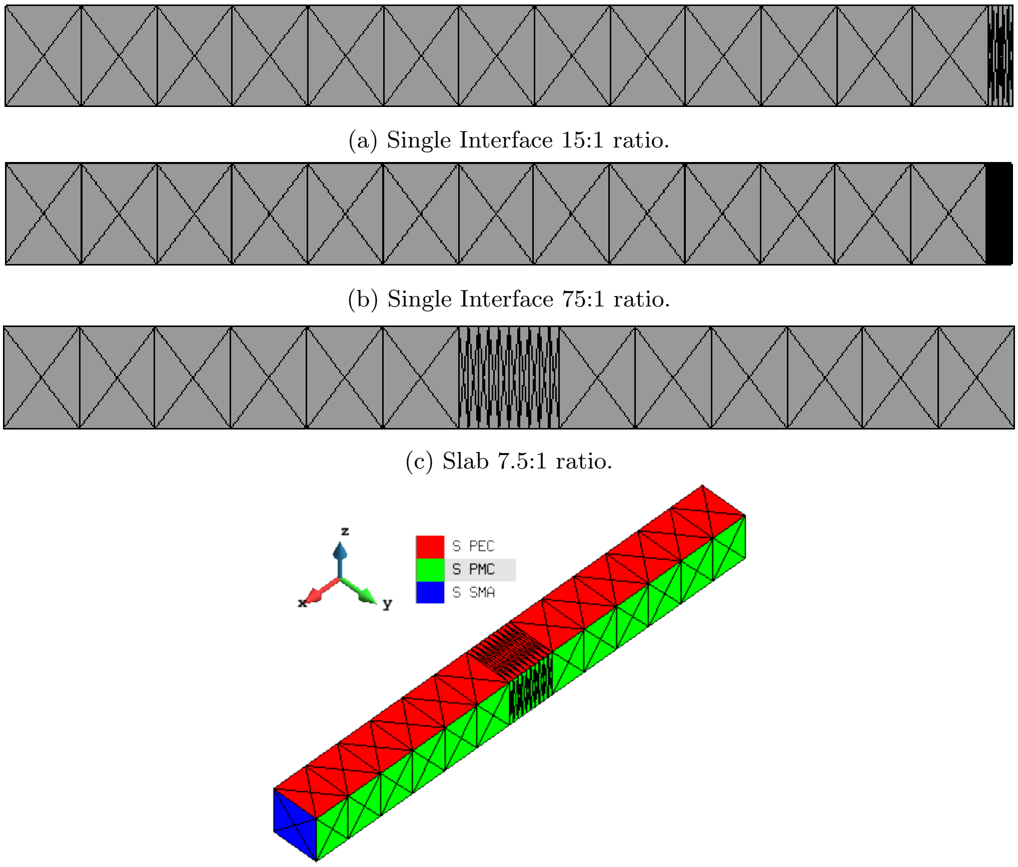
[1] Compiled with GNU C++ v4.6.3 using -O3 -ffast-math flags
[2] For more information visit: `http://www.ugrfdtd.es`
[3] For more information visit: `http://www.gidhome.com`

Algorithm 2: LSERK4-CPLTS algorithm. To evolve from $\mathbf{q}(t_n)$ to $\mathbf{q}(t_{n+1})$ this algorithm is called using (t_n), (Δt^{N_m-1}) and $(N_m - 1)$ as input arguments. The superscripts RHS and Res refer to Right Hand Side and Residue respectively. For the LSERK4, we have that $N_s = 5$, and the constants a_s, b_s and c_s are those shown in table 1.

```
**Data**: Inputs: $t$, $\Delta t$, $m$
**Result**: $\mathbf{q}_k(t)$ is evolved to $\mathbf{q}_k(t+\Delta t)$
$s \leftarrow 2$
while $s \leq N_s$ do
    if $m = N_m - 1$ then
        $k$ are all the elements in tier $m$.
    else
        $k$ are all the elements in tier $m$ and its $N_s - s$ neighbourhood.
    end
    $\mathbf{q}_k^{\mathrm{RHS}} \leftarrow \mathcal{L}_h(\mathbf{q}_k)$
    if $m > 0$ then
        $k^*$ are the elements in the 4-neighbourhood of $m - 1$.
        $\mathbf{q}_{k^*}^{\mathrm{Saved}} \leftarrow \mathbf{q}_{k^*}$
        Calls this algorithm with $(t + c_s\Delta t)$, $(h_s\Delta t)$ and $(m-1)$ as inputs.
        $\mathbf{q}_{k^*} \leftarrow \mathbf{q}_{k^*}^{\mathrm{Saved}}$
    end
    $\mathbf{q}_k^{\mathrm{Res}} \leftarrow a_s\mathbf{q}_k^{\mathrm{Res}} + \mathbf{q}_k^{\mathrm{RHS}}\Delta t$
    $\mathbf{q}_k \leftarrow \mathbf{q}_k + b_s\mathbf{q}_k^{\mathrm{Res}}$
    $s \leftarrow s + 1$
end
```

the ends of the computational domain and the other end is backed by an Silver-Mueller absorbing (SMA) boundary condition. The side–walls of the domain are Perfect Electric Conducting (PEC) and Perfect Magnetic Conducting (PMC) boundary conditions at the xy and xz planes respectively. The mesh is 1 m long from one end to the other. The coarse cell size is 7.5 cm and the cell sizes in the finer region vary from 0.1 to 0.5 cm.

Figures 5, 6 and 7 show the reflection coefficient in a range of frequencies. The closer the values are to zero the better are the properties of the scheme. When LF2 integration techniques are used, we observe two asymptotic regions, the steeper region corresponds to the range of frequency where the spatial error is dominating over the error coming from the order of the time integration. That is why no differences are observed between the different techniques used. At lower frequencies the time integration error dominates and different behaviours arise depending on the technique. We observe that for this case the LF2 with a fully defined state (LF2full) exhibits slightly better properties than the classic LF2 scheme. A possible explanation for this is that the incident wave is resolved using more time steps. In LF2-LTS and LF2-CPLTS, we observe some additional degradation when compared to the classic LF2 schemes. The CPLTS exhibits less reflection than the Montseny's LTS, the difference growing with the ratio between the coarser and finer mesh. With LSERK4 we observe that the spatial error dominates for the range of frequencies studied and the differences observed at lower frequencies can be attributed to errors



(a) Single Interface 15:1 ratio.

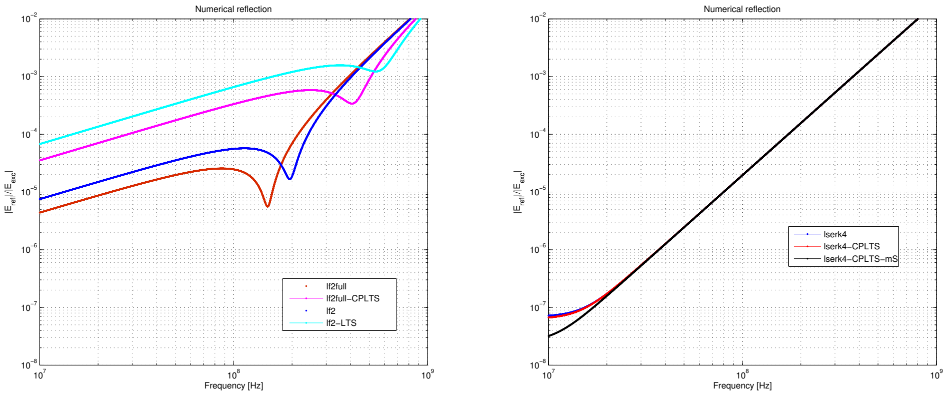
(b) Single Interface 75:1 ratio.

(c) Slab 7.5:1 ratio.

(d) Boundary conditions.

Figure 4: Meshes used for the study numerical reflections by an inhomogeneous mesh.

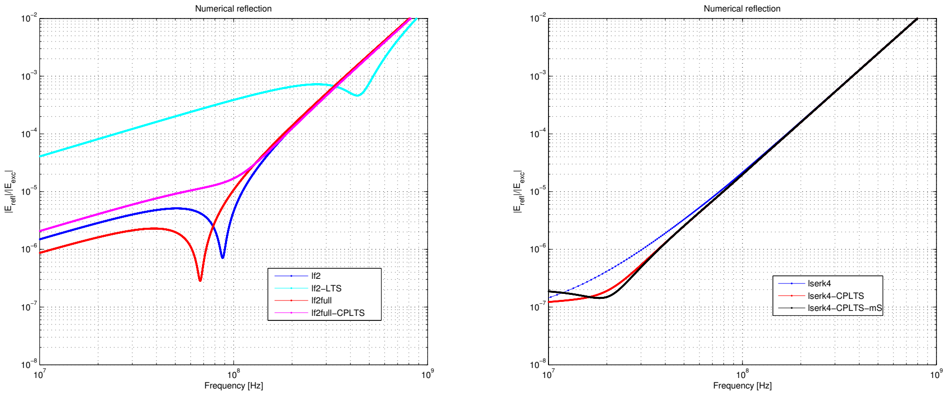
coming from illumination or poor absorption properties of the SMA boundary. The three LSERK4 figures exhibit a better behaviour than the LF2, as expected due to the higher order of the time integration technique, and for the cases studied, the different LSERK4 CPLTS implementations do not show a time integration error dominating over the spatial. When the maximum stage is used for the tier assortment, we observe a higher degradation in the low-frequency regime, probably because more time–stepping operations are being performed. The results for the LSERK4-CPLTS are very encouraging as we see little differences between the use the LSERK4-CPLTS technique and the classic LSERK4. Table 2 shows data corresponding to the tier assortment and computational times. As expected, the LF2-CPLTS is able of create more tiers than LF2-LTS because it only needs a ratio of two between maximum time step sizes. The CPU times for this simulation are listed for reference only and are not quite representative because the time employed to compute the excitation at the boundaries and the initialization is significant when compared with the operations performed to evolve the elements.



(a) LF2

(b) LSERK4

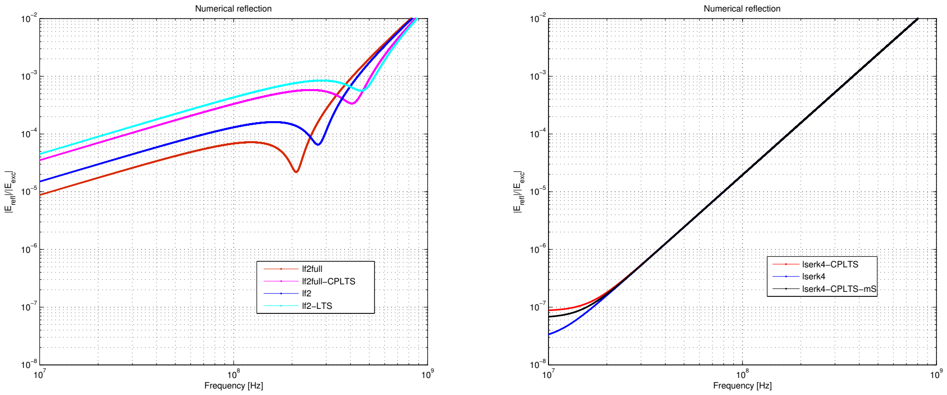
Figure 5: Numerical reflection from a single interface with ratio of 15:1



(a) LF2

(b) LSERK4

Figure 6: Numerical reflection from a single interface with ratio of 75:1



(a) LF2

(b) LSERK4

Figure 7: Numerical reflection from a slab with ratio of 7.5:1.

	Integrator	Number of Elements							Δt^m [ps]	CPU [s]
	Tier	0	1	2	3	4	5	6	0	
PW-Refl-r15-SInt	LSERK4-CPLTS	120	312	-	-	-	-	-	0.624	226
	LSERK4-CPLTS-mS	120	24	288	-	-	-	-	0.624	485
	LSERK4	432	-	-	-	-	-	-	0.624	468
	LF2-LTS	120	8	304	-	-	-	-	0.281	78
	LF2	432	-	-	-	-	-	-	0.281	211
	LF2full-CPLTS	80	48	12	292	-	-	-	0.281	173
	LF2full	432	-	-	-	-	-	-	0.281	1799
PW-Refl-r75-SInt	LSERK4-LTS	600	24	288	-	-	-	-	0.12	3148
	LSERK4-CPLTS-mS	600	24	24	264	-	-	-	0.12	15097
	LSERK4	912	-	-	-	-	-	-	0.12	4700
	LF2-LTS	600	8	12	292	-	-	-	0.06	1444
	LF2	912	-	-	-	-	-	-	0.06	2296
	LF2full-CPLTS	400	208	12	8	12	184	88	0.06	3524
	LF2full	912	-	-	-	-	-	-	0.06	7211
PW-Refl-r7.5-Slab	LSERK4-CPLTS	240	288	-	-	-	-	-	1.24	190
	LSERK4-CPLTS-mS	240	288	-	-	-	-	-	1.24	342
	LSERK4	528	-	-	-	-	-	-	1.24	325
	LF2-LTS	240	288	-	-	-	-	-	0.55	158
	LF2	528	-	-	-	-	-	-	0.55	151
	LF2full-CPLTS	160	96	272	-	-	-	-	0.55	157
	LF2full	528	-	-	-	-	-	-	0.55	254

Table 2: Element Tier assorting for LTS in the plane wave reflection.

5.2. *PEC cavity resonances*

As a second example we show comparisons of evolving a spatially uncorrelated random field (white noise) to study the resonances of a 1 m PEC cavity, in a similar way as done in [34]. The mesh used is depicted in Figure 4c with PEC boundaries at the ends rather than SMA. The resonance frequencies are obtained by performing the Fourier transform of the electric field evolution after 250 ns at a point separated 0.3m from one of the boundaries. Figure 8 show the eigenfrequencies obtained by the simulations together with the exact ones (black dashed vertical lines). The LF2 schemes do not show any particular difference with respect to their dispersive properties. The differences in amplitude between LF2 and LF2full can be attributed to the different initial treatment of fields. The LSERK4 schemes exhibit a similar behaviour in frequency but we observe additional attenuation when the CPLTS is used. When the tiers are assorted using the maximum stage criteria the attenuation is reduced. No late time stabilities were observed in any of the simulations. Figure 9 shows the discrete energy computed for both schemes. As we see, all the schemes present some dissipation coming from the upwinding. When the LSERK4-CPLTS technique is used we observe more dissipation if we do not use the max stage criteria for tier assortment. Figure 10 shows the error evolution at for an initial condition of three harmonics. In all cases we observe that the CPLTS introduces additional error when compared with the non LTS approaches. Table 3 shows data corresponding to the tier assortment and computational times. The CPU times show a clear improvement with the LSERK4-CPLTS algorithm while the gains for the LF2-LTS are more moderate. LF2-CPLTS does not perform better than the LF2.

	Integrator	Number of Elements					Δt^m [ps]	CPU [s]
	Tier	0	1	2	3	4	0	
reson-r7.5-Slab	LSERK4-CPLTS	240	288	-	-	-	1.24	2403
	LSERK4-CPLTS-mS	240	288	-	-	-	1.24	4051
	LSERK4	528	-	-	-	-	1.24	4013
	LF2-LTS	240	288	-	-	-	0.55	1207
	LF2	528	-	-	-	-	0.55	1917
	LF2full-CPLTS	160	96	272	-	-	0.55	2381
	LF2full	528	-	-	-	-	0.55	3615
rcs-1m	LSERK4-CPLTS	4535	57279	157	-	-	2.1	3733
	LSERK4	61971	-	-	-	-	2.1	8613
	LF2-LTS	522	8614	52798	37	-	0.95	963
	LF2	61971	-	-	-	-	0.95	4348
	LF2full-CPLTS	114	2155	7411	34521	17770	0.95	1851
	LF2full	61971	-	-	-	-	0.95	8642

Table 3: Element Tier assorting for LTS in the resonant cavity and RCS problems.

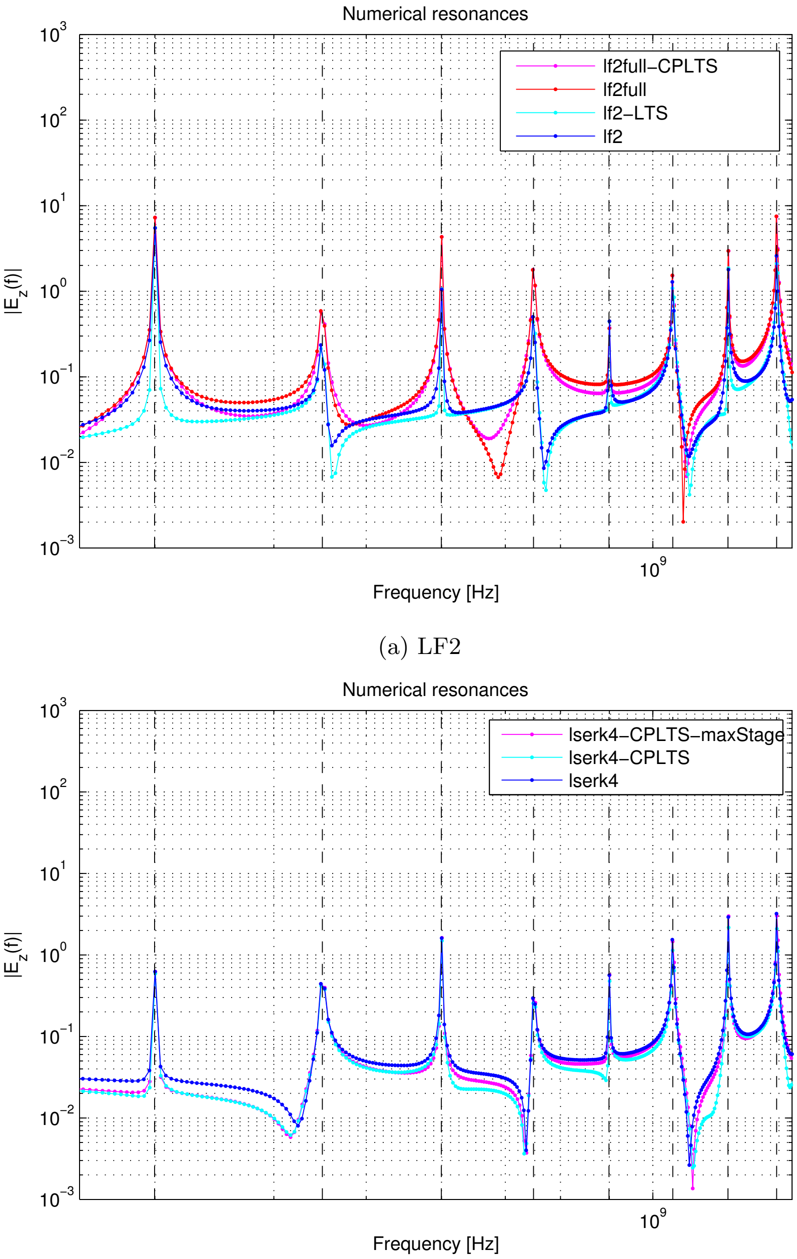
5.3. RCS Analysis of a PEC Sphere

As a last test case we present a bi–static Radar Cross Section (RCS) analysis [17]. Figure 11 show the boundary conditions used. Symmetry conditions were used to reduce the computational domain and the 1 m radius sphere was modelled using a PEC boundary condition. SMA boundary conditions were used to terminate the domain 3 m away from the surface of the sphere. The illumination was done using a Total Field/Scattered Field boundary condition in a spherical surface located 1 m away from the sphere using a Gaussian wave with 1 ns spread, y-polarization and propagating along the x axis. The typical element size of the mesh was 25 cm everywhere except in the PEC spherical surface modelling the sphere in which was set to 5 cm.

Figure 12 shows the results of the analysis for the various LF2 and LSERK4 schemes under study. At 450 MHz we see that the LF2 methods fit the Mie's analytical solution but the LF2 using Montseny's approach exhibits an angular offset caused by an appreciable difference in the dispersion relation. At 600MHz all methods present a higher deviation, caused by a poorer resolution of the spatial grid.

The LSERK4 results exhibit a better behaviour than the LF2, capturing the main features of the analytical solution. The application of CPLTS seems to better preserve the dispersion relation an thus the position of the peaks. However, at 600 MHz we can observe an appreciable numerical dissipation being introduced.

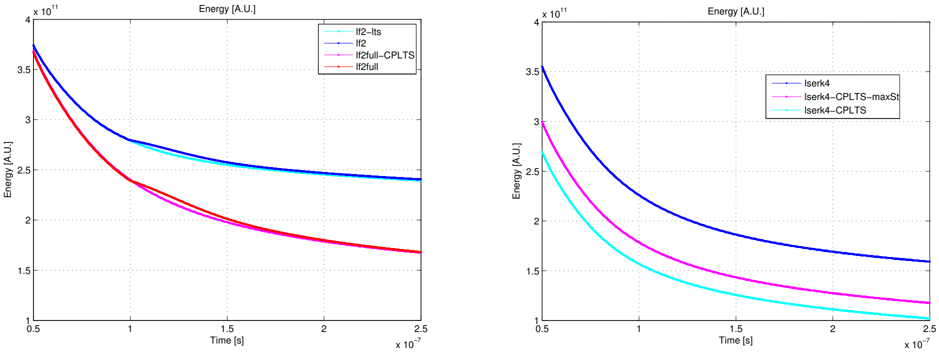
Table 3 shows data corresponding to the tier assortment and computational times. In this case, the LSERK4-CPLTS is able to provide a considerable speed up, reducing the CPU time from 8613 to 3733 s (~ 2). The LF2 LTS techniques yield a speed–up of about four times the non-LTS counterparts. The CPLTS speeds up the classic LF2 by a factor about two.



(a) LF2

(b) LSERK4

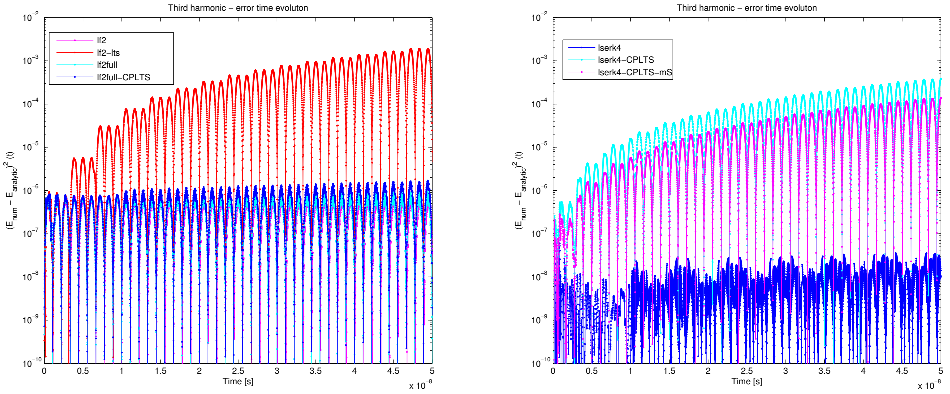
Figure 8: Resonances in a 1 m PEC cavity with slab meshing. Vertical dashed lines represent exact eigenfrequencies.



(a) LF2

(b) LSERK4

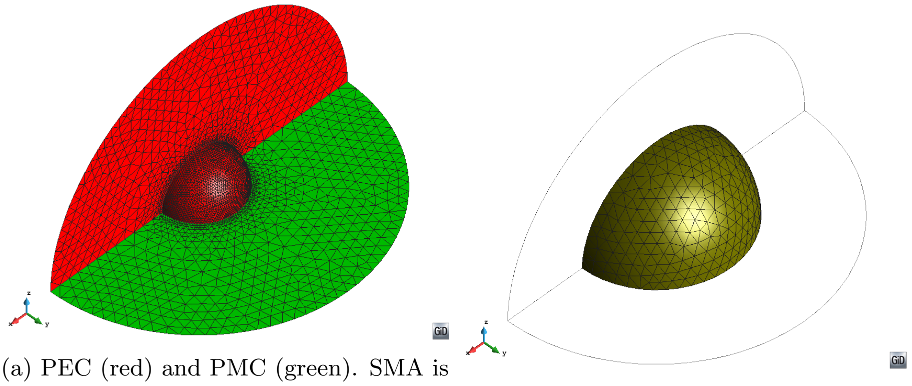
Figure 9: Energy evolution in a 1 m PEC cavity with slab meshing.



(a) LF2

(b) LSERK4

Figure 10: Evolution of the $(E_{\text{analytical}} - E_{\text{numerical}})^2$ error at the center of the 1 m PEC cavity with slab meshing for a three harmonics initial condition.



(a) PEC (red) and PMC (green). SMA is not depicted.

(b) Total Field region

Figure 11: Boundary conditions for the RCS case.

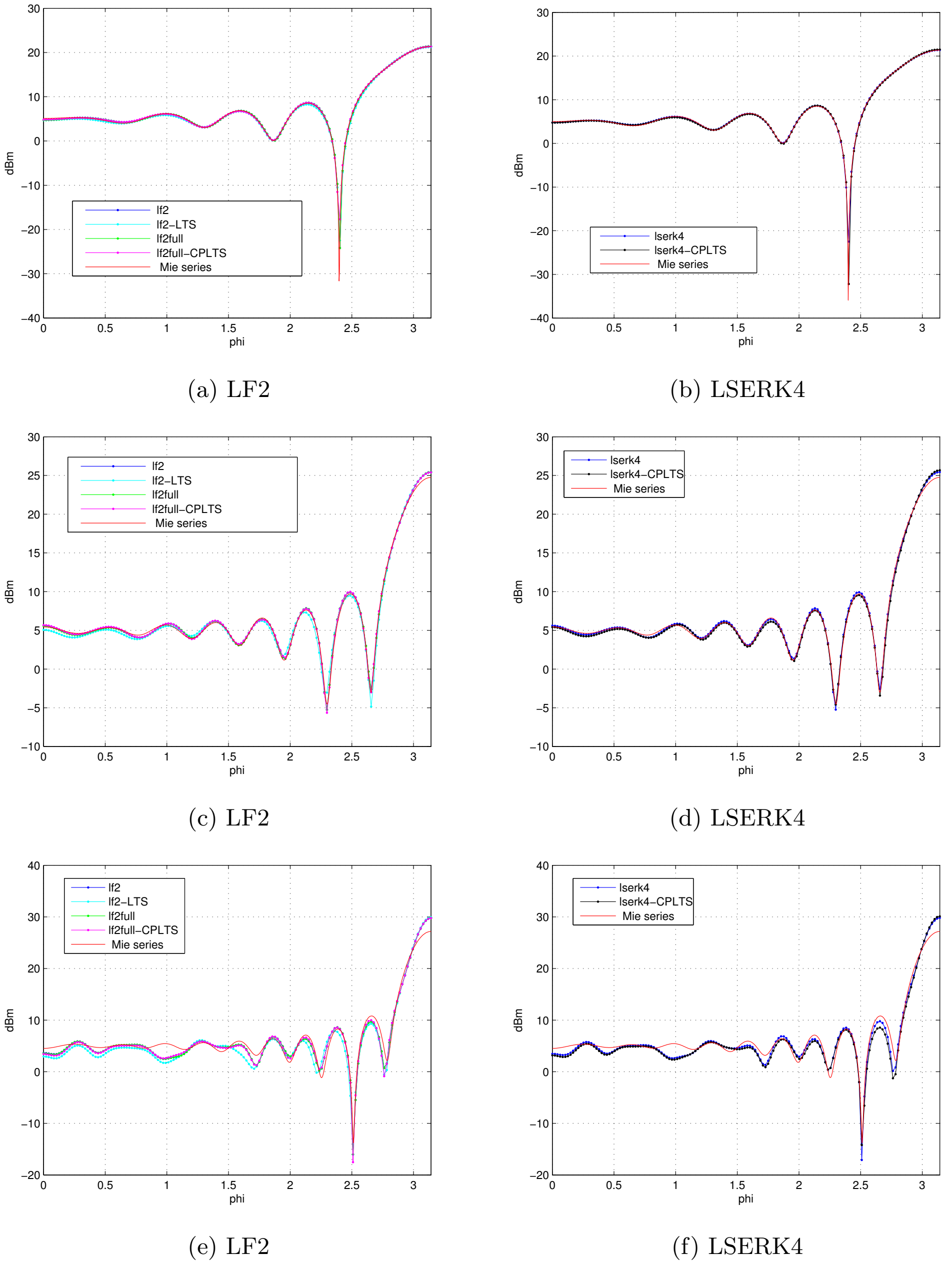
6. Tier assortment

In practice, an automated meshing process may produce a quite random tier assortment having an important impact in performance and accuracy. This occurs because we let the LTS algorithm and the tier–assortment to span the entire mesh. Notice that in practice this may not be necessary an optimal approach. Figures 13, 15, 16, 17, 18 and 19 illustrate this phenomenon. For the 1 m PEC sphere (Fig. 13, 15, 16), after imposing a constraint in the element size of 5 cm and leaving the rest with 25 cm we observe that there is an appreciable amount of scattered elements in the mesh belonging to a lower tier. The meshing algorithm is able to respect the sizes imposed to the elements in the regions closer to the surfaces but not in the inner part. Figures 17, 18 and 19 represent a variation of the 1m PEC sphere case in which an small cylinder representing a small scale feature has been appended to the sphere. In this example we observe that the presence of scattered lower tiers happens also in problems exhibiting disparate scales, unless the user pre-sets a given maximum number of tiers.

For the LSERK4 algorithm, scattered lower tiers degrade performance because, as depicted in Figure 14, many elements in the neighbourhood of lower tiers have to perform additional operations. Additionally, the CPLTS technique requires the storage of the elements in the neighbourhood of smaller tiers, increasing the memory consumption. Often the meshing and tier assorting processes result in the highest tier having a very small amount of elements (see Table 3), so it is up to the user whether to preserve those tiers or not. In the LF2-CPLTS case, we observe in Figures 15 and 18 that the assorting is able to create more tiers than in the LF2-LTS case. This has a positive impact in performance, which is specially relevant in cases with disparate spatial scales such as the presented in Figure 18.

7. Conclusions

In this work, we have introduced the Causal–Path concept as a way to perform LTS on explicit marching–in–time algorithms. We have applied this concept to the DG discretization under two different time integration techniques: LSERK4 and LF2.



(a) LF2

(b) LSERK4

(c) LF2

(d) LSERK4

(e) LF2

(f) LSERK4

Figure 12: Bi-static RCS at 300 MHz (top row), 450 MHz (middle row) and 600 MHz (bottom row). Continuous red line represents the analytical solution obtained through Mie's series.

When applied to LSERK4, the CPLTS implementation in which the tier assortment is done using the number of stages criterium has improved the performance by a factor of about two. For the case in which the assortment is done using the maximum stage size, no computational perfomance improvement has been observed but the numerical dissipation is reduced. For both cases the dispersive properties of the scheme do not seem to be significatively affected.

For LF2 the performance is also improved by a factor of about two for a bi–static RCS analysis case. In contrast, the commonly used Montseny's technique provides an speed up of about four. The CPLTS technique however seems to present better dispersive properties than the Montesny's approach and has better adaptivity to multiscale problems.

Acknowledgements

The work described in this paper and the research leading to these results have received funding from the European Community's Seventh Framework Programme FP7/2007-2013, under grant agreement no 205294 (HIRF SE project), and from the Spanish National Projects TEC2010-20841-C04-04, CSD2008-00068, and the Junta de Andalucia Project P09-TIC-5327. The work of FLT has been supported by U.S. NSF under grants 0925272 and 1305838.

8. Bibliograpy

[1] V. Dolean, H. Fahs, L. Fezoui, S. Lanteri, Locally implicit discontinuous galerkin method for time domain electromagnetics, J. Comput. Phys. 229 (2) (2010) 512–526. doi:10.1016/j.jcp.2009.09.038.
URL http://dx.doi.org/10.1016/j.jcp.2009.09.038

[2] S. Piperno, Symplectic local time-stepping in non-dissipative dgtd methods applied to wave propagation problems, ESAIM: Mathematical Modelling and Numerical Analysis 40 (5) (2007) 815–841.
URL http://eudml.org/doc/194337

[3] J. Alvarez, L. Angulo, A. Rubio Bretones, S. Garcia, A spurious-free discontinuous galerkin time-domain method for the accurate modeling of microwave filters, Microwave Theory and Techniques, IEEE Transactions on 60 (8) (2012) 2359–2369.

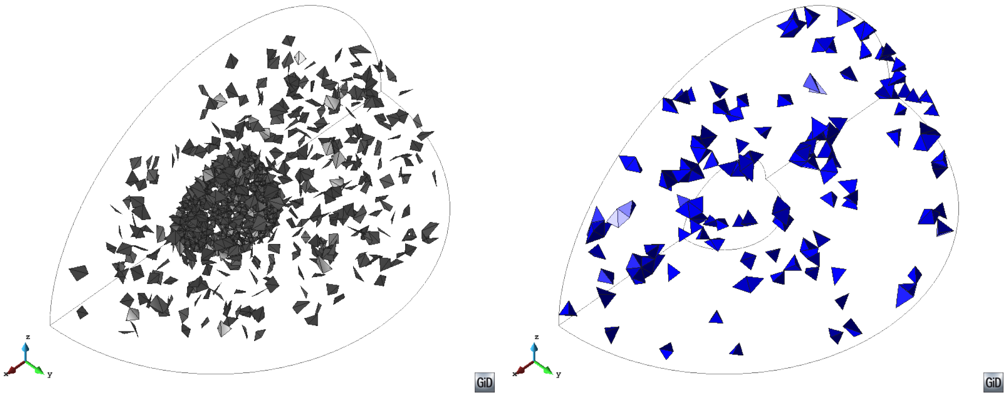
[4] L. Pebernet, X. Ferrieres, S. Pernet, F. Rogier, P. Degond, A discontinuous galerkin method to solve electromagnetic problems, IET Conference Publications 2008 (CP537) (2008) 96–97. doi:10.1049/cp:20080229.
URL http://link.aip.org/link/abstract/IEECPS/v2008/iCP537/p96/s1

[5] S. Dosopoulos, J.-F. Lee, Interior penalty discontinuous galerkin finite element method for the time-dependent first order maxwell's equations, Antennas and Propagation, IEEE Transactions on 58 (12) (2010) 4085 –4090. doi:10.1109/TAP.2010.2078445.

[6] E. Montseny, S. Pernet, X. Ferriéres, G. Cohen, Dissipative terms and local time-stepping improvements in a spatial high order discontinuous galerkin scheme for the time-domain maxwell's equations, J. Comput. Phys. 227 (14) (2008) 6795–6820. doi:10.1016/j.jcp.2008.03.032.
URL http://dx.doi.org/10.1016/j.jcp.2008.03.032

[7] J. Diaz, M. Grote, Energy conserving explicit local time stepping for second-order wave equations, SIAM Journal on Scientific Computing 31 (3) (2009) 1985–2014. arXiv:http://epubs.siam.org/doi/pdf/10.1137/070709414, doi:10.1137/070709414.
URL http://epubs.siam.org/doi/abs/10.1137/070709414

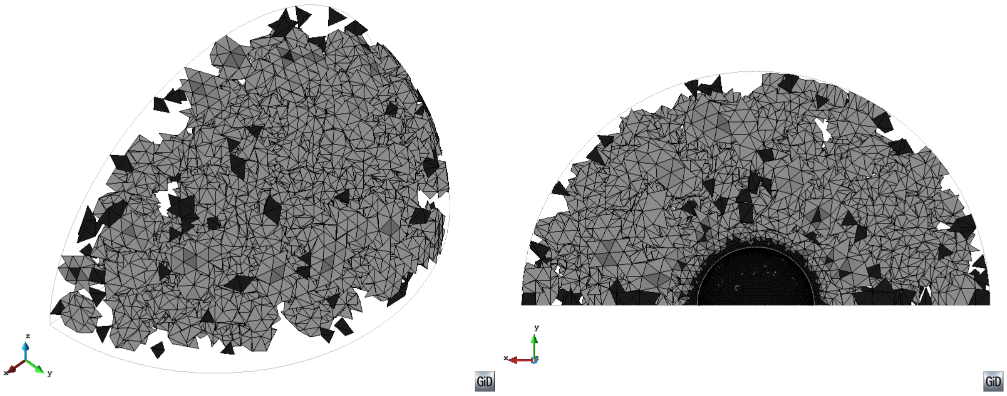
[8] M. J. Grote, T. Mitkova, Explicit local time-stepping methods for maxwells equations, Journal of Computational and Applied Mathematics 234 (12) (2010) 3283 – 3302. doi:10.1016/j.cam.2010.04.028.
URL http://www.sciencedirect.com/science/article/pii/S0377042710002360



(a) Tier 0

(b) Tier 2

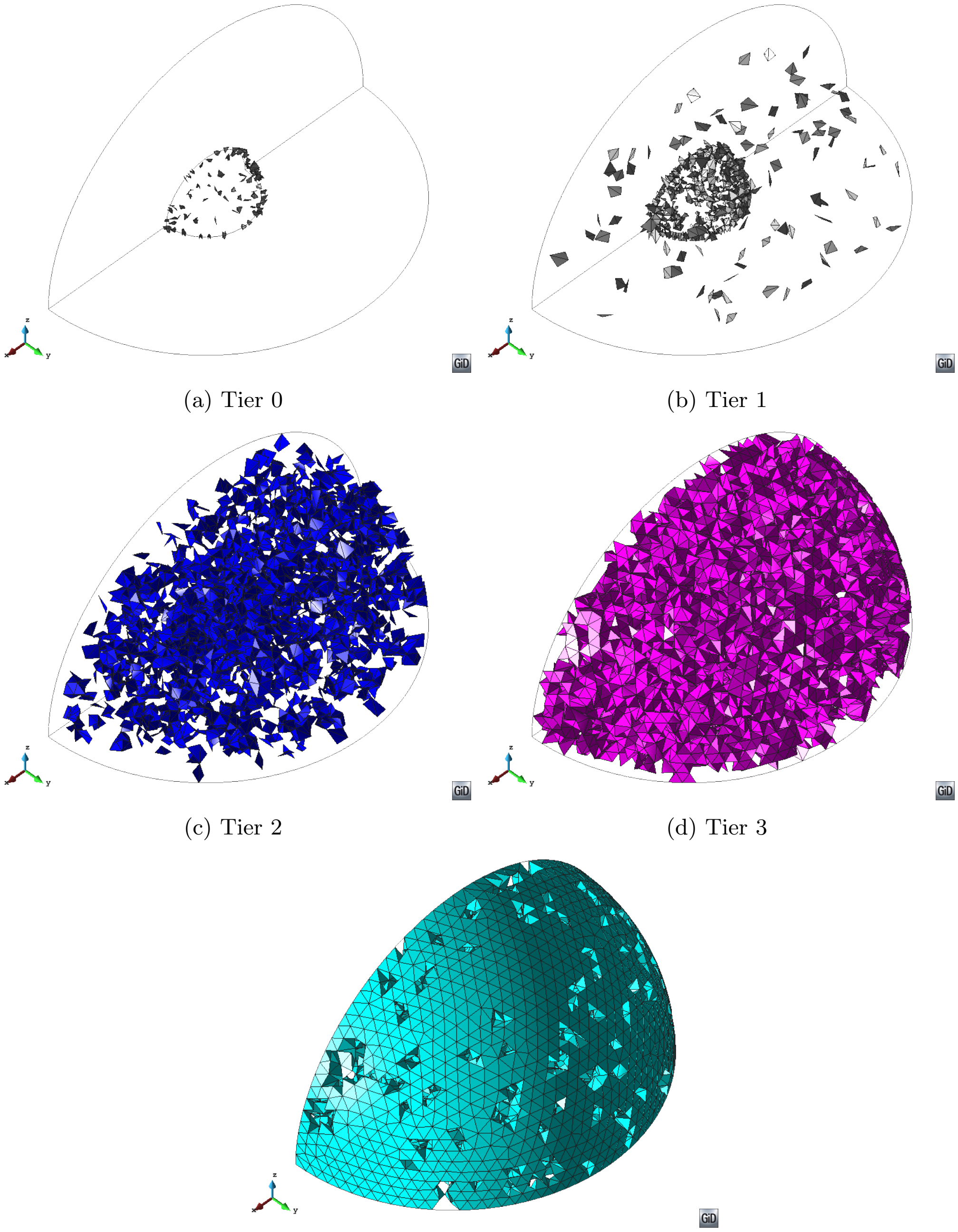
Figure 13: Tier assortment for LSERK4. Tier 1 is not represented.



(a) Stages distribution. Isometric.

(b) Stages distribution. XY plane.

Figure 14: Elements in LSERK4 where some operations are required by the smaller tiers. Darker colour means more operations (closer to a smaller tier). The degrees of freedom belonging to the elements represented need to be stored when the smaller tier is solved. Elements that do not require additional operations and storage are not represented.



(a) Tier 0

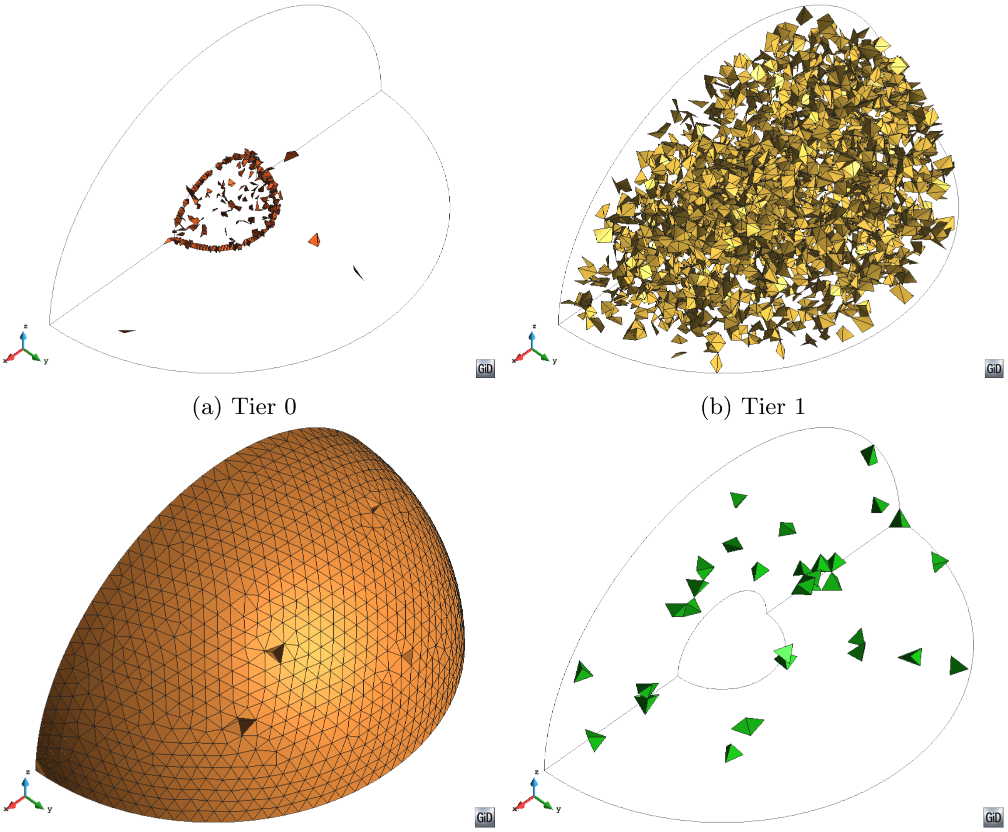
(b) Tier 1

(c) Tier 2

(d) Tier 3

(e) Tier 4

Figure 15: Tier assortment for LF2-CPLTS.



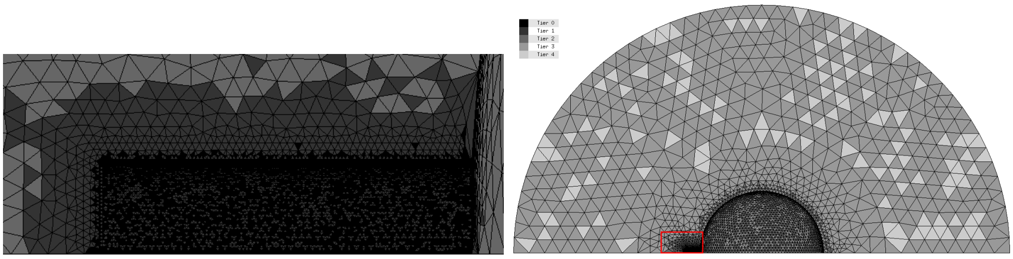
(a) Tier 0

(b) Tier 1

(c) Tier 2

(d) Tier 3

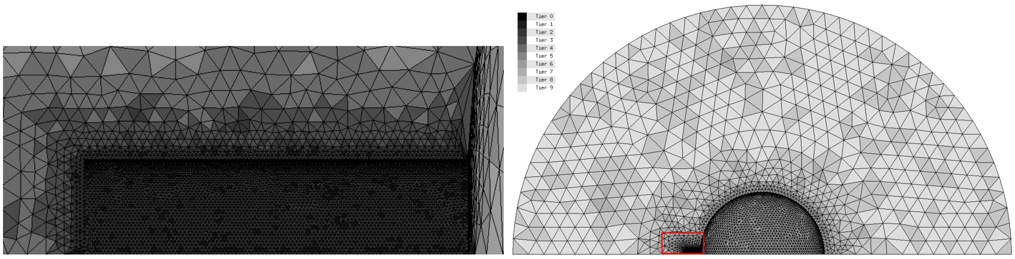
Figure 16: Tier assortment for LF2-LTS.



(a) LSERK4-CPLTS Detail view

(b) LSERK4-CPLTS General view

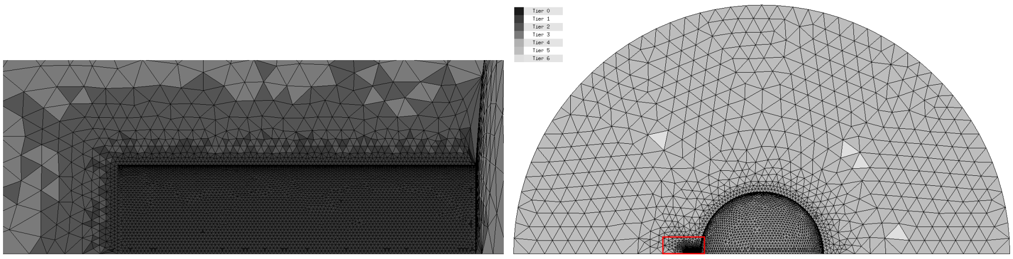
Figure 17: Tier assortment for LSERK4-CPLTS



(a) LF2-CPLTS Detail view

(b) LF2-CPLTS General view

Figure 18: Tier assortment for LF2-CPLTS



(a) LF2-LTS Detail view

(b) LF2-LTS General view

Figure 19: Tier assortment for LF2-LTS

[9] M. Kaser, M. Dumbser, J. de la Puente, H. Igel, An arbitrary high order discontinuous galerkin method for elastic waves on unstructured meshes iii: Viscoelastic attenuation, Geophysical Journal International 168 (1) (2007) 224–242. doi:10.1111/j.1365-246X.2006.03193.x.
URL http://www.geophysik.uni-muenchen.de/~igel/PDF/kaeseretal{_}gji{_}2006.pdf

[10] D. Sarmany, M. Botchev, J. van der Vegt, Time-integration methods for finite element discretisations of the second-order maxwell equation, Computers & Mathematics with Applications 65 (3) (2013) 528 – 543. doi:10.1016/j.camwa.2012.05.023.
URL http://www.sciencedirect.com/science/article/pii/S0898122112004713

[11] B. Zhao, J. Young, S. Gedney, Spice lumped circuit subcell model for the discontinuous galerkin finite-element time-domain method, Microwave Theory and Techniques, IEEE Transactions on 60 (9) (2012) 2684 –2692. doi:10.1109/TMTT.2012.2203923.

[12] J. Alvarez, L. D. Angulo, A. R. Bretones, S. G. Garcia, A comparison of the fdtd and lfdg methods for the estimation of hirf transfer functions, in: Proceedings on Computational ElectroMagnetics And Electromagnetic Compatibility 2013 (CEMEMC13), 2013.

[13] J. Alvarez, L. D. Angulo, A. R. Bretones, C. M. D. J. V. Coevorden, S. G. Garcia, An efficient dgtd method including local time-stepping: application to antenna modeling, IEEE Trans. on Antennas and PropagationSubmitted.

[14] J. Alvarez, L. D. Angulo, M. Cabello, A. R. Bretones, S. G. Garcia, An analysis of the leap-frog discontinuous galerkin method for maxwell equations, IEEE Transactions on Antennas and Propagation.Submitted.

[15] J. S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, 1st Edition, Springer Publishing Company, Incorporated, 2007.

[16] D. Sarmany, M. A. Botchev, J. J. Vegt, Dispersion and dissipation error in high-order runge-kutta discontinuous galerkin discretisations of the maxwell equations, J. Sci. Comput. 33 (2007) 47–74. doi:10.1007/s10915-007-9143-y.
URL http://portal.acm.org/citation.cfm?id=1288603.1288623

[17] J. Alvarez, L. Angulo, M. Fernandez Pantoja, A. Rubio Bretones, S. Garcia, Source and boundary implementation in vector and scalar dgtd, Antennas and Propagation, IEEE Transactions on 58 (6) (2010) 1997 –2003. doi:10.1109/TAP.2010.2046857.

[18] L. D. Angulo, J. Alvarez, S. Garcia, A. R. Bretones, R. G. Martin, Discontinuous galerkin time-domain method for gpr simulation of conducting objects, Near Surface Geophysics 9 (2011) 257–263.

[19] J. Niegemann, W. Pernice, K. Busch, Simulation of optical resonators using dgtd and fdtd, Journal of Optics A: Pure and Applied Optics 11 (11) (2009) 114015.
URL http://stacks.iop.org/1464-4258/11/i=11/a=114015

[20] R. Diehl, K. Busch, J. Niegemann, Comparison of low-storage runge-kutta schemes for discontinuous galerkin time-domain simulations of maxwell's equations, Journal of Computational and Theoretical Nanoscience 7 (8) (2010) 1572–1580. doi:doi:10.1166/jctn.2010.1521.
URL http://www.ingentaconnect.com/content/asp/jctn/2010/00000007/00000008/art00035

[21] H. Fahs, High-order leap-frog based discontinuous galerkin method for the time-domain maxwell equations on non-conforming simplicial meshes, Numerical Mathematics: A Journal of Chinese Universities 2 (3) (2009) 275 – 300.
URL http://hal.archives-ouvertes.fr/hal-00600467/en/

[22] C. Chauviere, J. Hesthaven, A. Kanevsky, T. Warburton, High-order localized time integration for grid-induced stiffness, in: K. Bathe (Ed.), Computational Fluid and Solid Mechanics 2003, Elsevier Science Ltd, Oxford, 2003, pp. 1883 – 1886. doi:http://dx.doi.org/10.1016/B978-008044046-0.50461-9.
URL http://www.sciencedirect.com/science/article/pii/B9780080440460504619

[23] A. Taube, M. Dumbser, C.-D. Munz, R. Schneider, A high-order discontinuous galerkin method with time-accurate local time stepping for the maxwell equations, International Journal of Numerical Modelling: Electronic Networks, Devices and Fields 22 (1) (2009) 77–103. doi:10.1002/jnm.700.
URL http://dx.doi.org/10.1002/jnm.700

[24] L. Liu, X. Li, F. Q. Hu, Nonuniform time-step runge kutta discontinuous galerkin method for computational aeroacoustics, Journal of Computational Physics 229 (19) (2010) 6874 – 6897. doi:10.1016/j.jcp.2010.05.028.
URL http://www.sciencedirect.com/science/article/pii/S0021999110002895

[25] C.-W. Shu, Discontinuous galerkin methods: General approach and stability, lecture notes.

[26] S. Sherwin, Dispersion analysis of the continuous and discontinuous galerkin formulations., in: in International Symposium on Discontinuous Galerkin Methods, Springer, 1999, pp. 425–431.

[27] J. Alvarez, L. Angulo, A. Rubio Bretones, S. Gonzalez Garcia, 3d discontinuous galerkin time–

domain method for anisotropic materials, Antennas and Wireless Propagation Letters, IEEE 11 (2012) 1182–1185.

[28] J. S. Hesthaven, T. Warburton, Highorder nodal discontinuous galerkin methods for the maxwell eigenvalue problem, Philosophical Transactions of the Royal Society of London. Series A:Mathematical, Physical and Engineering Sciences 362 (1816) (2004) 493–524. arXiv:http://rsta.royalsocietypublishing.org/content/362/1816/493.full.pdf+html, doi:10.1098/rsta.2003.1332.
URL http://rsta.royalsocietypublishing.org/content/362/1816/493.abstract

[29] S. Dosopoulos, Interior penalty discontinuous galerkin finite element method for the time-domain maxwell's equations, Ph.D. thesis, Ohio State University (2012).

[30] J. H. Williamson, Low-storage runge-kutta schemes, Journal of Computational Physics 35 (1) (1980) 48 – 56. doi:DOI:10.1016/0021-9991(80)90033-9.
URL http://www.sciencedirect.com/science/article/B6WHY-4DD1NRP-H1/2/a1a09857d87c5a3538d4800fd648be47

[31] R. Otin, Regularized maxwell equations and nodal finite elements for electromagnetic field computations, Electromagnetics 30 (2010) 190–204.
URL http://prod.informaworld.com/10.1080/02726340903485489

[32] R. A. Chilton, H-, p- and t-refinement strategies for the finite-difference-time-domain (fdtd) method developed via finite-element (fe) principles, Ph.D. thesis, Graduate School of The Ohio State University (2008).

[33] B. Donderici, F. Teixeira, Conformal perfectly matched layer for the mixed finite element time-domain method, Antennas and Propagation, IEEE Transactions on 56 (4) (2008) 1017 –1026. doi:10.1109/TAP.2008.919215.

[34] J. Kim, F. Teixeira, Parallel and explicit finite-element time-domain method for maxwell's equations, Antennas and Propagation, IEEE Transactions on 59 (6) (June) 2350–2356. doi:10.1109/TAP.2011.2143682.