An exact particle splitting algorithm for PIC codes

M. Drouin\textsuperscript{a,}\textsuperscript{*}

\textsuperscript{a}Sorbonne Universit\'e, Observatoire de Paris, Universit\'e PSL, CNRS, LERMA, F-75252, Paris, France

Abstract

Splitting and merging are long standing issues in PIC codes. I propose a novel algorithm devoted to exact splitting for Particle-In-Cell (PIC) codes relying on Adaptive Mesh Refinement (AMR) grids. AMR grids have - by definition - a constant refinement factor between levels, this strong property combined with B-spline shape functions makes possible exact splitting of macro-particles in space. The proposed exact splitting method is applicable for any integer refinement factor and B-spline shape functions of arbitrary order. Besides, this exact algorithm removes by construction the usual numerical noise increase resulting from approximate particle splitting strategies.

Keywords: PIC; MLMD; AMR; particle; exact splitting; numerical noise; Particle-In-Cell

1. Introduction

In Particle-In-Cell (PIC) codes, a large collection of particles is described by a smaller set of macro-particles, each one being representative of a large number of physical particles. Such a model is acceptable as long as the statistical properties of the macro-particles are close to those of the physical particles.

In plasma physics, the multi-scale character of the systems to be studied is ubiquitous. As a non-exhaustive list, we can mention turbulence in magnetized plasmas [1], magnetic reconnection [2], shocks induced by supersonic/superalfvenic flows [3]. Furthermore, each of these mechanisms can

\textsuperscript{*}Corresponding author.

E-mail address: drouin.mathieu@gmail.com
have strong connections between them \[4\]. Thereby, the smallest scale to resolve imposes constraints on the grid size, whereas the largest imposes constraints on the simulation domain. Satisfying both conditions may eventually result computationally prohibitive. In situations where small-scales are spatially localized, computational efficiency can be achieved by using Adaptive Mesh Refinement (AMR) grids or Moving Mesh Adaptation (MMA). In addition to these mesh techniques, it might also be necessary to increase or decrease the number of particles depending on local accuracy requirements \[5\].

Generally speaking, splitting and/or merging methods are required for two main reasons in PIC codes relying on structured meshes. First, it is desirable to control accuracy by locally increasing or reducing the number of macro-particles - in order to get a better description of plasma dynamics and to reduce numerical noise. Second, spurious self-force effects which break momentum conservation appear at boundaries between two regions with different resolution.

Let us briefly review the different strategies related to splitting/merging methods in adaptive PIC codes.

The idea of the Moving Mesh Adaptation (MMA) approach is to change the mesh size while conserving the particle size in index space. When using this approach, we can implicitly defines a time and space dependant refinement ratio. This method natively induces a term $\partial S/\partial t$ in the equations of motion \[6\]. Practically, this error term is made small by ensuring the change of the particle shape during motion is small enough ($\partial S/\partial t << 1$).

Within the MMA-PIC approach, splitting/merging schemes (also called particle rezoning schemes) \[5\] are used only to monitor the local accuracy.

It is possible to design AMR-PIC methods without splitting the macro-particles. However, a macro-particle is submitted to a self-force when it transitions between two meshes with different resolutions. Different methods were proposed to reduce self-force errors at refinement boundaries either in the electrostatic case \[7\] or in the electromagnetic case \[8\]. Within the AMR-PIC approach, splitting/merging schemes provide means to avoid self-force effects at the cost of a dynamic management of the particle population.

The essential difference between the MLMD-PIC \[9, 10, 11\] and AMR-PIC approaches stems from particle management. In MLMD-PIC particles are evolved on each domain (from the coarsest to the finest AMR level) whereas in AMR-PIC they are only evolved on the most refined level. Hence, the splitting process on AMR grids either replaces a mother particle by daughter particles.
particles [12, 13, 14], or is used to initialize/repopulate a refined patch within
the MLMD framework [9].
To our knowledge, in all AMR or MLMD PIC codes (e.g. [12, 14, 9]) macro-
particles are split using heuristic methods which only conserve grid moments
approximately. My main motivation is to propose an exact numerical scheme
to handle splitting properly in AMR and MLMD PIC codes. As the MLMD
method only requires a splitting algorithm, the exact method further dis-
cussed in this paper is of particular interest in this context.
This paper is based on the observation that a constant refinement ratio
between levels is used to build AMR grids. Therefore we can benefit the
property - B-spline basis functions are refinable - that is a B-spline basis
function defined on a coarse mesh can be represented as a linear combination
of B-spline basis functions on a refined mesh [15, 16]. In most PIC codes, the
assignment functions used are B-splines therefore exact splitting in space is
possible as will be further presented. Let us recall that no additional numer-
ical noise is induced whenever the spatial particle distribution resulting from
the splitting process is strictly equivalent to the original distribution. The
exact method satisfies the aforementioned condition, therefore no additional
numerical noise is induced on refined patches.
Let us underline that approximate particle splitting methods violate equiv-
alence between original and split particle distributions. Yet, approximate
methods always produce more numerical noise on the refined patch where
split particles are deposited - when compared to the numerical noise induced
by the parent particle distribution.
Section 2 is devoted to general conditions under which two sets of particles
are considered equivalent. In section 3, we derive the theoretical equations
linking a B-spline function at the coarse level to the corresponding linear
combination at a refined level. We give special focus on the most frequent
configurations encountered in PIC codes, namely a refinement factor equal
to 2 or 3 with an arbitrary high spline order, and next an arbitrary refine-
ment factor but only for 1st order spline function. In section 4, we provide
the basic steps of the splitting algorithm needed for inclusion in a PIC code.
In section 5, we briefly address the 2D and 3D generalization of the present
method. Finally in Section 6, the ability of the present algorithm to accu-
rately reproduce a coarse moment on a refined mesh is demonstrated.
2. General considerations

The problem of particle rezoning can be formulated as the replacement of a set of \( N \) particles with position \( x_p \), velocity \( v_p \), charge \( q_p \), and mass \( m_p \), with a different set of \( N' \) particles with position \( x_{p'} \), velocity \( v_{p'} \), charge \( q_{p'} \), and mass \( m_{p'} \). The two sets of particles are considered physically equivalent under the following conditions:

- The two sets are indistinguishable on the basis of their contributions to the grid moments.
- The two sets of particles sample the same velocity distribution function.

The first criterion requires the two sets of particles to give the same moments of the particle distribution. These moments are used to solve the field equations, if this first criterion is satisfied exactly total energy and momentum are also automatically conserved. The second criterion is automatically verified when we restrict to spatial splitting. It may be violated when splitting is done in velocity space, which is out of the scope of the present paper.

2.1. Notations

Let us denote the reduced coordinate \( \xi \), the refinement factor \( \alpha \), and \( \mu \) the interpolation order. The reduced coordinate is defined by \( \xi = x / \Delta x_L \), where \( x \) denotes the spatial coordinate and \( \Delta x_L \) is the spatial step size of the base mesh.

We also define

\[
\xi_{g_p}^{L_0} = \frac{x_g - x_p}{\Delta x_L},
\]

where \( x_g \) and \( x_p \) respectively denote a grid point coordinate, and a particle coordinate. Using the definition of \( \alpha \) we have the relation

\[
\xi_{g'}^{L_1} = \frac{x_{g'} - x_{p'}}{\Delta x_L} = \alpha \xi_{g'}^{L_0}.
\]

2.2. Contribution to the grid moments in general

We now examine the first criterion in details. For the sake of simplicity we restrict theoretical development in this section to 1D (without loss of generality).
The 1D moments are defined at the grid points \( x_g \) using the reduced coordinates \( \xi_g = \xi(x_g) \) and \( \xi_{gp}^{L0} \) previously introduced, we get

\[
M(\xi_g) = \frac{1}{\Delta x_{L0}} \sum_p S(\xi_{gp}) q_p F(v_p). \tag{3}
\]

The function \( F \) of the particle velocity characterizes the moment. In explicit PIC codes, only the charge density is required \( (F(v_p) = 1) \)

\[
\rho(\xi_g) = \frac{1}{\Delta x_{L0}} \sum_p S(\xi_{gp}) q_p. \tag{4}
\]

In electromagnetic PIC codes, the current density is also used \( (F(v_p) = v_p) \)

\[
J(\xi_g) = \frac{1}{\Delta x_{L0}} \sum_p S(\xi_{gp}) q_p v_p. \tag{5}
\]

In moment implicit PIC and hybrid PIC methods, the pressure tensor is required \( (F(v_p) = v_p v_p) \)

\[
\Pi(\xi_g) = \frac{1}{\Delta x_{L0}} \sum_p S(\xi_{gp}) q_p v_p v_p. \tag{6}
\]

To summarize, for each required moment we must satisfy relations of the form

\[
\sum_p F(v_p) S(\xi_{gp}) q_p = \sum_{p'} F(v_{p'}) S(\xi_{gp'}) \varepsilon_{p'q_p}, \tag{7}
\]

where the unknowns are the coefficients \( \varepsilon_{p'} \). We note that the current and pressure tensor equivalence equations between the two sets of particles are vectorial whether no hypothesis is made on \( F(v_p) \) and \( F(v_{p'}) \). Besides \( [7] \) defines a requirement for equivalence between two sets of particles, either when splitting or merging is considered. In the next subsection, we will derive a simplified version of \( [7] \) in the splitting case.

### 2.3. Contribution to the grid moments when splitting

Here, we consider a single particle for a specified species \( (q_p = q_{p'}) = q \) with velocity \( v_1 \) which is split into \( n \) new particles (with velocities \( v_{1'} \)). Hence the equivalence equation becomes

\[
S(\xi_{gp}) = \sum_{p'} \varepsilon_{p'} S(\xi_{gp'}). \tag{8}
\]
Thereby we obtain a general equivalence between an initial set of particles and a final set of particles (after the split operation) by only ensuring that each individual split operation obeys (8). Practically, Eq. (8) is written for two consecutive refinement levels \( L_0 \) and \( L_1 \)

\[
S(\xi_{g_{p'}}^{L_0}) = \sum_{p'} \varepsilon_{p'} S(\xi_{g_{p'}}^{L_1}),
\]

where \( \xi_{g_{p'}}^{L_0} \) and \( \xi_{g_{p'}}^{L_1} \) are respectively defined on \( L_0 \) and \( L_1 \) grids. Using (2) we finally get

\[
S(\xi_{g_{p'}}^{L_0}) = \sum_{p'} \varepsilon_{p'} S(\alpha \xi_{g_{p'}}^{L_0}).
\]

In the next section, we will derive \( \varepsilon_{p'} \) so that (10) is exactly satisfied when the assignment function is a B-spline which is the most common case in PIC codes. To simplify further theoretical developments we will use translated transforms of the centred spline function of degree \( \mu \) (i.e. \( S_\mu \)), these transforms are denoted by \( B_\mu \). The reader interested by the explicit definitions of \( S_\mu \) is referred to Appendix A.

3. Refinement properties of B-splines

A remarkable property of B-splines is that they obey a refinement equation. This specific property paves the way to connect splines and subdivision ([16], sec 2.2.3, p. 25). The B-Spline of degree 0, defined by

\[
B_0(\xi) = \begin{cases} 1 & \text{if } 0 \leq \xi \leq 1 \\ 0 & \text{elsewhere} \end{cases}
\]

(11)

can be written as a sum of translated and contracted transforms of itself

\[
B_0(\xi) = B_0(2\xi) + B_0(2\xi - 1).
\]

(12)

And the B-spline of degree \( l \) is defined by

\[
B_l(\xi) = \bigotimes_{i=0}^{l} B_0(\xi),
\]

(13)

where \( \otimes \) denotes the continuous convolution operator. This operator is classically defined for two functions \( f(t) \) and \( g(t) \) by \( (f \otimes g)(t) = \int_{-\infty}^{\infty} f(t - \tau) g(\tau) \, d\tau \).
\[ g(t) = \int f(s)g(t - s)ds. \]

Prior to developing \( B_1 \), let us recall basic properties of convolution for functions \( f, g \) and \( h \)

\[
\begin{align*}
  f(t) \otimes (g(t) + h(t)) &= f(t) \otimes g(t) + f(t) \otimes h(t) \quad (14) \\
  f(t - i) \otimes g(t - k) &= m(t - i - k) \quad (15) \\
  f(2t) \otimes g(2t) &= \frac{1}{2}m(2t) \quad (16)
\end{align*}
\]

3.1. Arbitrary spline order with \( \alpha = 2 \)

\( B_1 \) can be expressed as a sum of contracted \( B_1 \) functions. This can be derived using (13), and writing \( B_0 \) as a sum with (12). Thereby, we get

\[
B_1(\xi) = B_0(\xi) \otimes B_0(\xi) = (B_0(2\xi) + B_0(2\xi - 1)) \otimes (B_0(2\xi) + B_0(2\xi - 1)) \quad (17)
\]

\[
= \frac{1}{2} (B_1(2\xi) + 2B_1(2\xi - 1) + B_1(2\xi - 2)) \quad (18)
\]

\[
= \frac{1}{2^2} \sum_{k=0}^{2} \binom{2}{k} B_1(2\xi - k) \quad (19)
\]

When \( \alpha = 2 \), we can prove the following relation

\[
B_{\mu}(\xi) = \frac{1}{2^\mu} \sum_{k=0}^{\mu+1} \binom{\mu + 1}{k} B_{\mu}(2\xi - k), \quad (21)
\]

where \( \binom{\mu + 1}{k} \) denotes the binomial coefficients \( \binom{\mu + 1}{k} = \frac{(\mu + 1)!}{k!(\mu + 1 - k)!} \).

3.2. Arbitrary spline order with \( \alpha = 3 \)

In this case the B-Spline of degree 0 is defined by

\[
B_0(\xi) = B_0(3\xi) + B_0(3\xi - 1) + B_0(3\xi - 2). \quad (22)
\]
Figure 1: Decomposition of a B-Spline from 1st to 3rd order with a refinement factor $\alpha = 2$.

$B_1$ can be formally derived using [13], we get

$$B_1(\xi) = B_0(\xi) \otimes B_0(\xi)$$

$$= (B_0(3\xi) + B_0(3\xi - 1) + B_0(3\xi - 2)) \otimes (B_0(3\xi) + B_0(3\xi - 1) + B_0(3\xi - 2))$$

$$= \frac{1}{3} [B_1(3\xi) + B_1(3\xi - 1) + B_1(3\xi - 2) +$$

$$+ B_1(3\xi - 1) + B_1(3\xi - 2) + B_1(3\xi - 3) +$$

$$+ B_1(3\xi - 2) + B_1(3\xi - 3) + B_1(3\xi - 4)]$$

$$= \frac{1}{3} [B_1(3\xi) + 2B_1(3\xi - 1) + 3B_1(3\xi - 2)$$

$$+ 2B_1(3\xi - 3) + B_1(3\xi - 4)]$$

(23)

When the refinement factor is equal to 3, we can prove by recurrence the following relation

$$B_\mu(\xi) = \frac{1}{3^\mu} \sum_{k=0}^{2(\mu+1)} \binom{\mu+1}{k} B_\mu(3\xi - k)$$

(24)

Where $\binom{n}{k}_2$ corresponds to the trinomial coefficients, i.e. the coefficients of the following polynome:

$$P(x) = (1 + x + x^2)^n.$$  

(25)

As in the binomial case, a triangle can also be drawn in the trinomial case
\[ n = 0: \quad 1 \]
\[ n = 1: \quad 1 \quad 1 \quad 1 \]
\[ n = 2: \quad 1 \quad 2 \quad 3 \quad 2 \quad 1 \]
\[ n = 3: \quad 1 \quad 3 \quad 6 \quad 7 \quad 6 \quad 3 \quad 1 \]
\[ n = 4: \quad 1 \quad 4 \quad 10 \quad 16 \quad 19 \quad 16 \quad 10 \quad 4 \quad 1 \]

Figure 2: Decomposition of a B-Spline from 1st to 3rd order with a refinement factor \( \alpha = 3 \).

3.3. Arbitrary spline order with arbitrary \( \alpha \in \mathbb{Z}^* \)

The general case is ruled by the following formula

\[
B_\mu(\xi) = \frac{1}{\alpha^\mu} \sum_{k=0}^{(\alpha-1)(\mu+1)} \binom{\mu + 1}{k} B_{\mu}(\alpha\xi - k) \quad (26)
\]

Numerical computation of the multinomial coefficients becomes computationally intensive when \((\alpha - 1) > 2\). Whenever a large value of the refinement factor is to be considered \((i.e. \ alpha > 3)\) with a second or higher order spline, these coefficients should be precomputed and stored in a table.

3.4. Special case: arbitrary \( \alpha \in \mathbb{Z}^* \) with \( \mu = 1 \) (piecewise linear)

In general, computing the multinomial coefficients can be quite puzzling. But, we note that the 2nd line of a multinomial triangle is straightforward

\[ n = 2: \quad 1 \quad 2 \quad \ldots \quad \alpha - 1 \quad \alpha \quad \alpha - 1 \quad \ldots \quad 2 \quad 1 \]
This is of major interest for the piecewise linear case as we can write the base function as a combination of contracted transforms with arbitrary refinement factor. In this particular case, the base function reads

\[ B_1(\xi) = \frac{1}{\alpha} \sum_{k=0}^{2(\alpha-1)} \binom{2}{k}_{\alpha-1} B_1(\alpha \xi - k) \]  

(27)

\[ 0.0 \quad 0.5 \quad 1.0 \quad 1.5 \quad 2.0 \]
\[ 0.0 \quad 0.2 \quad 0.4 \quad 0.6 \quad 0.8 \quad 1.0 \]

\[ S(\xi) \]

(a) \( \mu = 1, \alpha = 2 \)  
(b) \( \mu = 1, \alpha = 3 \)  
(c) \( \mu = 1, \alpha = 5 \)

Figure 3: Decomposition of a 1st order B-Spline for different refinement factors.

4. Practical steps of the splitting algorithm

Each macro-particle to be split is processed as follows

- Compute the number of children particles

\[ N_c = [(\alpha - 1)(\mu + 1) + 1]^\text{dim}, \]  

(28)

where \text{dim} stands for the space dimensionality.

- Compute the corresponding weights

\[ w_k = \frac{1}{\alpha^\mu} \binom{\mu + 1}{k}_{\alpha-1} \]  

(29)

with \( k = 0, 1 \ldots N_c - 1 \)

- Compute the total weight

\[ w_{\text{tot}} = \sum_{k=0}^{N_c-1} w_k \]  

(30)
• Compute the positions of children particles
  
  We define the minimum index

  \[ i x_{\text{min}} = -\frac{(N_c - 1)}{2}. \]  

The positions of the new particles are given by

\[ x_k = x_0 + \delta x_k, \]  

where \( x_0 \) denotes the position of the mother particle and

\[ \delta x_k = (i x_{\text{min}} + k) \times \Delta x L1 \]

are the offsets.

• Create the children particles
  
  The position, weight and velocity of the new particles are defined by

\[ x_k = x_0 + \delta x_k, \]  

\[ W_k = W_0 \times \frac{w_k}{w_{\text{tot}}}, \]  

\[ V_k = V_0, \]  

where the mother position, weight and velocity are respectively denoted by \( x_0, W_0 \) and \( V_0 \).

Implementation examples of the numerical schemes described in section 3 are available online: https://github.com/droudrou/exact_splitting

5. Extension to 2D and 3D

Extension to 2D and 3D is straightforward as long as multi-dimensional weight factors are obtained with tensorial products of 1D spline functions. The number of split particle when \( \alpha = 2 \) is summarized in Tab. as highlighted the exact splitting method becomes prohibitive in 3D when using a high order interpolation function. For example, each parent particle will be split into 64 new particles if a 2nd order interpolation function is used.
Table 1: Number of daughter particles as function of the Spline order ($\mu$) and the dimensionality, for a refinement factor $\alpha = 2$. For simplicity, we recall the formula to get the number of daughter particles $N_c = [(\alpha - 1)(\mu + 1) + 1]^{\text{dim}}$.

| $\mu$ | 1 | 2 | 3 | 4 |
|-------|---|---|---|---|
| dim   | 1 | 3 | 4 | 5 | 6 |
| 2     | 9 | 16| 25| 36|
| 3     | 27| 64| 125| 216|

6. Exact splitting verification

The algorithm described in this paper ensures that the particle distribution loaded on a refined patch - when it is created - will be strictly equivalent to the particle distribution of the coarse (i.e. parent) patch. To do this verification we initialize a Maxwellian plasma in a simulation box comprising a refined patch. The simulation setup consists in a 1D grid with 512 cells, $\Delta x = 0.1$, periodic boundary conditions and an arbitrary time step. The thermal velocity and density of the considered species are set respectively to $v_{th} = 0.01$, $n_s = 1$. The Maxwellian distribution of the considered species is discretized with 100 particles per cell on the coarse domain. Next, the refined patch is initialized with coarse domain particle information using the splitting algorithm described in section 4.

As expected the charge density initialized on the refined patch (Figs. 4 in green) is superimposed on the coarse charge density curve (Figs. 4 in blue), the latter property is satisfied using any interpolation function from 1st to 4th order. Equivalently, these results prove that no additional numerical noise is induced when using exact splitting.
Figure 4: Charge density after initialization ($t = 0$) using interpolation functions from $1^{st}$ to $4^{th}$ order. The numerical noise is given by the standard deviation of the represented field. High order interpolation reduces the initial level of numerical noise, as evidenced when comparing the noise level between Fig. 4(a) and Fig. 4(d). The coarse and refined patch are depicted in blue and green respectively. The plasma is discretized with 100 particles per cell.
7. Conclusion

As already mentioned, in a MLMD code particles are evolved on each domain. Blocks of refined mesh therefore have their own particle distribution that does not replace the underlying coarse particle distribution, thereby particle merging is not required. Yet, splitting is still necessary to populate a refined patch. In this context, an exact splitting method might be essential to maintain numerical noise to an acceptable level especially in 2D and 3D configurations.

In the present paper, I provide an exact particle splitting algorithm applicable with reasonable computational cost in 1D, 2D and 3D (at least with 1st order interpolation). This splitting scheme can be used with a spline interpolation function of arbitrary order and any refinement factor. Practically, it is well suited to arbitrary high refinement factor when a first order interpolation function (i.e. piece-wise linear case) is used. Moreover, it is a very convenient way to suppress additional numerical noise induced by approximate splitting strategies when populating a refined patch. Finally, we verified the accuracy of the exact splitting method - as expected coarse moments are reproduced with no error at the refined level.
Appendix A. Analytical shape functions

The typical shape function used in a PIC code to achieve the moment deposition and field interpolation steps is a B-spline. Let us denote $S_k$ the centered shape function of degree $k$. $S_k$ functions are just translated transforms of the base functions $B_k$ previously introduced, we have

$$
S_0(\xi) = B_0(\xi + 1/2) \\
S_1(\xi) = B_1(\xi + 1) \\
S_2(\xi) = B_2(\xi + 3/2) \\
\ldots \\
(A.1)
$$

Next, we provide the explicit definition of the centered shape functions.

Appendix A.1. 0th order

$$
S_0(\xi) = \begin{cases} 
1 & \text{if } -1/2 \leq \xi \leq 1/2 \\
0 & \text{elsewhere}
\end{cases} \\
(A.2)
$$

Appendix A.2. 1st order

$$
S_1(\xi) = \begin{cases} 
\xi + 1 & \text{if } -1 \leq \xi \leq 0 \\
1 - \xi & \text{if } 0 \leq \xi \leq 1 \\
0 & \text{elsewhere}
\end{cases} \\
(A.3)
$$

Appendix A.3. 2nd order

$$
S_2(\xi) = \begin{cases} 
\frac{1}{2} \left( \frac{3}{2} + \xi \right)^2 & \text{if } -3/2 \leq \xi \leq -1/2 \\
\frac{3}{4} - \xi^2 & \text{if } -1/2 \leq \xi \leq 1/2 \\
\frac{1}{2} \left( \frac{3}{2} - \xi \right)^2 & \text{if } 1/2 \leq \xi \leq 3/2 \\
0 & \text{elsewhere}
\end{cases} \\
(A.4)
$$

Appendix A.4. 3rd order

$$
S_3(\xi) = \begin{cases} 
\frac{4}{3} \left( 1 + \frac{\xi}{2} \right)^3 & \text{if } -2 \leq \xi \leq -1 \\
-\frac{\xi^3}{2} - \xi^2 + 2/3 & \text{if } -1 \leq \xi \leq 0 \\
\frac{\xi^3}{2} - \xi^2 + 2/3 & \text{if } 0 \leq \xi \leq 1 \\
\frac{4}{3} (1 - \frac{\xi}{2})^3 & \text{if } 1 \leq \xi \leq 2 \\
0 & \text{elsewhere}
\end{cases} \\
(A.5)
$$
Appendix A.5. 4th order

\[ S_4(\xi) = \begin{cases} 
\frac{1}{24} (\frac{5}{2} + \xi)^4 & \text{if } -\frac{5}{2} \leq \xi \leq -\frac{3}{2} \\
\frac{1}{90} (55 - 20\xi - 120\xi^2 - 80\xi^3 - 16\xi^4) & \text{if } -\frac{3}{2} \leq \xi \leq -\frac{1}{2} \\
\frac{1}{192} (115 - 120\xi^2 + 48\xi^4) & \text{if } -\frac{1}{2} \leq \xi \leq \frac{1}{2} \\
\frac{1}{96} (55 + 20\xi - 120\xi^2 + 80\xi^3 - 16\xi^4) & \text{if } \frac{1}{2} \leq \xi \leq \frac{3}{2} \\
\frac{1}{24} (\frac{5}{2} - \xi)^4 & \text{if } \frac{3}{2} \leq \xi \leq \frac{5}{2} \\
0 & \text{elsewhere}
\end{cases} \]

(A.6)

References

[1] T. Parashar, S. Servidio, B. Breech, M. Shay, W. Matthaeus, Kinetic driven turbulence: structure in space and time, Physics of Plasmas (1994-present) 17 (10) (2010) 102304.

[2] R. M. Kulsrud, Intuitive approach to magnetic reconnection, Physics of Plasmas (1994-present) 18 (11) (2011) 111201.

[3] J. Giacalone, J. R. Jokipii, Magnetic field amplification by shocks in turbulent fluids, The Astrophysical Journal Letters 663 (1) (2007) L41.

[4] H. Karimabadi, V. Roytershteyn, H. Vu, Y. Omelchenko, J. Scudder, W. Daughton, A. Dimmock, K. Nykyri, M. Wan, D. Sibeck, et al., The link between shocks, turbulence, and magnetic reconnection in collisionless plasmas, Physics of Plasmas (1994-present) 21 (6) (2014) 062308.

[5] G. Lapenta, Particle rezoning for multidimensional kinetic particle-in-cell simulations, Journal of Computational Physics 181 (1) (2002) 317 – 337. doi:http://dx.doi.org/10.1006/jcph.2002.7126.

[6] G. Lapenta, DEMOCRITUS: An adaptive particle in cell (PIC) code for object-plasma interactions, Journal of Computational Physics 230 (12) (2011) 4679 – 4695. doi:http://dx.doi.org/10.1016/j.jcp.2011.02.041.

[7] P. Colella, P. C. Norgaard, Controlling self-force errors at refinement boundaries for amr-pic, Journal of Computational Physics 229 (4) (2010) 947 – 957. doi:http://dx.doi.org/10.1016/j.jcp.2009.07.004.
[8] J.-L. Vay, D. Grote, R. Cohen, A. Friedman, Novel methods in the particle-in-cell accelerator code-framework warp, Computational Science & Discovery 5 (1) (2012) 014019.

[9] M. Innocenti, G. Lapenta, S. Markidis, A. Beck, A. Vapirev, A multi level multi domain method for particle in cell plasma simulations, Journal of Computational Physics 238 (2013) 115 – 140. doi:http://dx.doi.org/10.1016/j.jcp.2012.12.028

[10] A. Beck, M. E. Innocenti, G. Lapenta, S. Markidis, Multi-level multi-domain algorithm implementation for two-dimensional multiscale particle in cell simulations, Journal of Computational Physics 271 (2014) 430–443.

[11] M. Innocenti, A. Beck, S. Markidis, G. Lapenta, Momentum conservation in multi-level multi-domain (mlmd) simulations, Journal of Computational Physics 312 (2016) 14–18.

[12] K. Fujimoto, S. Machida, Electromagnetic full particle code with adaptive mesh refinement technique: Application to the current sheet evolution, Journal of Computational Physics 214 (2) (2006) 550 – 566. doi:http://dx.doi.org/10.1016/j.jcp.2005.10.003

[13] K. Fujimoto, R. D. Sydora, Electromagnetic particle-in-cell simulations on magnetic reconnection with adaptive mesh refinement, Computer Physics Communications 178 (12) (2008) 915 – 923. doi:http://dx.doi.org/10.1016/j.cpc.2008.02.010

[14] J. Müller, S. Simon, U. Motschmann, J. Schüle, K.-H. Glassmeier, G. J. Pringle, A.I.K.E.F.: Adaptive hybrid model for space plasma simulations, Computer Physics Communications 182 (4) (2011) 946 – 966. doi:http://dx.doi.org/10.1016/j.cpc.2010.12.033

[15] P. Bornemann, F. Cirak, A subdivision-based implementation of the hierarchical b-spline finite element method, Computer Methods in Applied Mechanics and Engineering 253 (2013) 584 – 598. doi:http://dx.doi.org/10.1016/j.cma.2012.06.023

[16] D. Zorin, P. Schröder, Subdivision for Modeling and Animation : SIGGRAPH 2000 Course Notes (2000).