ANNALS OF GEOPHYSICS, 54, 5, 2011; doi: 10.4401/ag-5344

Special Issue: V3-LAVA PROJECT

Moving least-squares corrections for smoothed particle hydrodynamics

Giuseppe Bilotta[1,2], Giovanni Russo[1,*], Alexis Hérault[2,3], Ciro Del Negro[2]

[1] *Università di Catania, Dipartimento di Matematica e Informatica, Catania, Italy*

[2] *Istituto Nazionale di Geofisica e Vulcanologia, Sezione di Catania, Osservatorio Etneo, Catania, Italy*

[3] *Conservatoire des Arts et Métiers, Département Ingénierie Mathématique, Paris, France*

Article history
Received November 21, 2010; accepted July 15, 2011.
Subject classification:
Algorithms and implementation, Smoothed particle hydrodynamics, Moving least-squares, Mesh-free.

ABSTRACT

First-order moving least-squares are typically used in conjunction with smoothed particle hydrodynamics in the form of post-processing filters for density fields, to smooth out noise that develops in most applications of smoothed particle hydrodynamics. We show how an approach based on higher-order moving least-squares can be used to correct some of the main limitations in gradient and second-order derivative computation in classic smoothed particle hydrodynamics formulations. With a small increase in computational cost, we manage to achieve smooth density distributions without the need for post-processing and with higher accuracy in the computation of the viscous term of the Navier–Stokes equations, thereby reducing the formation of spurious shockwaves or other streaming effects in the evolution of fluid flow. Numerical tests on a classic two-dimensional dam-break problem confirm the improvement of the new approach.

1. Introduction

Smoothed particles hydrodynamics (SPH) is a meshless Lagrangian method where a fluid is discretized using virtual interpolation nodes (particles) that are not fixed in any predefined mesh and where their evolution is described by the equation of motion of the fluid itself. The advantages of SPH include direct computation of all required physical quantities and their gradients, implicit tracking of surfaces (such as the free surface of the flow or internal solidification fronts in the case of thermal evolutions with phase transition), and the ability to cope with large deformations without any need for remeshing (which is typically needed, for example, with the finite elements method).

SPH was originally developed in the late 1970s by Gingold and Monaghan [1977] and Lucy [1977], with applications relating to astrophysics. Recently SPH has gained a lot of attention in other fields of computational fluid dynamics, with important applications ranging from coastal engineering [Hérault et al. 2009, Dalrymple and Hérault 2009] to geophysics and volcanology [Hérault et al. 2010b, 2011]. These applications rely heavily on the ease with which SPH can model both the mechanical and thermal evolution of free-surface fluids. These extensive applications have also highlighted some of the limitations that standard SPH formulations encounter, in terms of accuracy or when fluids with high viscosity are modeled.

The purpose of this paper is to present an alternative approach that while keeping close to the spirit of SPH, improves on the general accuracy of the method, and provides a stronger foundation for better evaluation of the higher order derivatives that are necessary for the viscous term of classic fluid-modeling equations.

1.1. Standard smoothed particles hydrodynamics interpolation for fields

The mathematical foundation for SPH lies in the properties of convolutions. For any scalar field u in a domain $\Omega \subseteq \mathbf{R}^d$, by definition of the Dirac delta distribution δ we can write, with a typical abuse of notation:

$$u(\mathbf{x}) = \int_\Omega u(\mathbf{y})\delta(\mathbf{x}-\mathbf{y})d\mathbf{y}.$$

We can approximate δ with a family of *smoothing kernels* $W(\cdot, h)$, parameterized by their *smoothing length* h, satisfying:

$$\int_{\mathbf{R}^d} W(\mathbf{x},h)d\mathbf{x} = 1$$

$$\lim_{h\to 0} W(\cdot,h) = \delta$$

where the limit is intended in the sense of the distributions. We then have:

$$u(\mathbf{x}) \simeq \int_\Omega u(\mathbf{y})W(\mathbf{x}-\mathbf{y},h)d\mathbf{y}.$$

Physically, the domain Ω represents the body of the fluid, which we want to discretize with a finite set of *particles* with given masses m_i, densities ρ_i and volumes $V_i = m_i/\rho_i$. Formally, let a density function ρ: $\Omega \to \mathbf{R}$ and a set of points

at locations $\{\mathbf{x}_i\} \subset \Omega$ be given. We approximate ρ in the weak sense with a discretization:

$$\rho(\mathbf{x}) = \int_\Omega \rho(\mathbf{y})\delta(\mathbf{x}-\mathbf{y})d\mathbf{y} \simeq \simeq \int_\Omega \rho(\mathbf{y})W(\mathbf{x}-\mathbf{y},h)d\mathbf{y} \simeq \sum_i m_i W(\mathbf{x}-\mathbf{x}_i,h)$$

where the summation is extended to all of the particles and the values m_i are called the *masses* of the particles. The weak approximation is read to mean that for any field u defined in Ω, we can write:

$$\rho(\mathbf{x})u(\mathbf{x}) = \int_\Omega \rho(\mathbf{y})u(\mathbf{y})\delta(\mathbf{x}-\mathbf{y})d\mathbf{x} \simeq \simeq \sum_i m_i u(x_i) W(\mathbf{x}-\mathbf{x}_i,h)$$

Hence:

$$u(\mathbf{x}) = \frac{u(\mathbf{x})}{\rho(\mathbf{x})}\rho(\mathbf{x}) \simeq \sum_i \frac{u(x_i)}{\rho_i} m_i W(\mathbf{x}-\mathbf{x}_i,h) = = \sum_i u(x_i) V_i W(\mathbf{x}-\mathbf{x}_i,h)$$

where $\rho_i = \rho(\mathbf{x}_i)$ and $V_i = m_i/\rho_i$. This gives us the standard SPH interpolation for field values:

$$< u(\mathbf{x}) > = \sum_i u(x_i) V_i W(\mathbf{x}-\mathbf{x}_i,h)$$

The smoothing kernels are usually chosen with compact support, so that the summation is extended only to particles in a *neighborhood* of $\mathbf{x}$.

We note that there are two errors that contribute to the SPH interpolation. The first is related to the approximation of δ with the smoothing kernel W with length h, and goes to zero as $h \to 0$. The second source of error in the SPH interpolation is given by the space discretization, and therefore it depends on the particle density or equivalently from the average inter-particle spacing H. Again, the continuous limit must be obtained for $H \to 0$, and to ensure that the entire SPH method is consistent, we should have $H/h \to 0$ as $h \to 0$, although some different approaches have been devised to circumvent this issue, such as the renormalizing mesh-free scheme of Lanson and Vila [2008]. In applications, the average inter-particle distance H is usually a fixed fraction of the smoothing length h; typically $h = \alpha H$ with $2 < \alpha < 4$, with the actual value depending on the smoothing kernel used.

The accuracy of the interpolation is related to the number of moments of W that are zero; this is always guaranteed for first moments as long as the kernel is chosen center-symmetric, such that $W(\cdot, h)$ only depends on $|\mathbf{x}-\mathbf{x}_i|$.

To simplify the notation in what follows, we will write $r_i = |\mathbf{x}-\mathbf{x}_i|$, $r_{ij} = |\mathbf{x}_i-\mathbf{x}_j|$ and W_{ij} as a short form for $W(r_{ij}, h)$. Kernel symmetry ensures that $W_{ij} = W_{ji}$. We observe that for center-symmetric kernels, we can compute their gradient as:

$$\nabla_{\mathbf{x}} W(\mathbf{x}-\mathbf{x}_i,h) = \frac{\mathbf{x}-\mathbf{x}_i}{r_i}\frac{\partial W(r,h)}{\partial r}\bigg|_{r=r_i}. \quad (1)$$

When choosing an SPH kernel, it is often convenient to select one for which the factor:

$$F(r,h) = \frac{1}{r}\frac{\partial W(r,h)}{\partial r} \quad (2)$$

can be computed analytically without an actual division by r. This ensures that the kernel gradient does not suffer from singularities from overlapping particles.

SPH kernels are also usually chosen as positive, so that the interpolated density:

$$< \rho(\mathbf{x}) > = \sum_i m_i W(\mathbf{x}-\mathbf{x}_i,h)$$

is always strictly positive. This condition, however, prevents the second moments of W from being zero, and effectively limits the accuracy of the SPH interpolation to first order at best.

An important limitation of SPH is that when the interpolation is computed on the location of a particle, the interpolated value is generally different from the interpolating field value. We have:

$$< u(\mathbf{x}_j) > = \sum_i u_i W_{ij} V_i \neq u_j$$

for an arbitrary field u.

We say that a numerical interpolation scheme has order k consistency when the interpolated values and the interpolating values match on the interpolation nodes for all of the polynomials of degree at most k. A significant limit of SPH is that in its standard formulation, it does not guarantee even order 0 consistency, meaning that even constant fields are not reconstructed exactly. This is usually remedied by introducing a *corrected* kernel:

$$\tilde{W}(r_j,h) = \frac{W(r_j,h)}{\sum_i W_{ij} V_i}$$

(the Shepard correction) that ensures the exact interpolation of constant functions. It should be noted, however, that while the Shepard correction gives a lower minimum and average relative error, it can give a higher maximum error than standard SPH for nonconstant fields. We shall see in the next section that a correction based on moving least-squares (MLS) can be used to guarantee a higher order consistency of SPH.

1.2. Smoothed particles hydrodynamics for first derivatives

For gradients, SPH takes advantage of the properties of convolutions and the divergence theorem to transfer the ∇ operator on the kernel. We have:

$$\begin{aligned}
&\nabla u(\mathbf{x}) = \\
&= \int_\Omega \nabla_y u(\mathbf{y})\delta(\mathbf{x}-\mathbf{y})d\mathbf{y} = \\
&= \int_\Sigma u(\mathbf{y})\delta(\mathbf{x}-\mathbf{y})\mathbf{n}d\Sigma - \int_\Omega u(\mathbf{y})\nabla_y\delta(\mathbf{x}-\mathbf{y})d\mathbf{y} = \\
&= \int_\Sigma u(\mathbf{y})\delta(\mathbf{x}-\mathbf{y})\mathbf{n}d\Sigma + \int_\Omega u(\mathbf{y})\nabla_x\delta(\mathbf{x}-\mathbf{y})d\mathbf{y} \simeq \\
&\simeq \int_\Sigma u(\mathbf{y})W(\mathbf{x}-\mathbf{y},h)\mathbf{n}d\Sigma + \int_\Omega u(\mathbf{y})\nabla_x W(\mathbf{x}-\mathbf{y},h)d\mathbf{y}
\end{aligned}$$

where $\Sigma = \partial\Omega$ and $\mathbf{n}$ is its outer normal.

If the compact support of W does not intersect Σ, the first integral is zero, and this leads us to the classic first form of the SPH interpolation of the gradient:

$$< \nabla u(\mathbf{x}) > = \sum_i u_i V_i \nabla W(r_i, h), \tag{3}$$

that is first order accurate in h as long as the point $\mathbf{x}$ is far from the boundary of Ω.

Near the boundary, the surface integral $\int_\Sigma u(\mathbf{y})\delta(\mathbf{y}-\mathbf{x})d\Sigma$ is nonzero. A number of strategies have been devised to compensate for this term for particles near the physical boundaries of the domain [Liu and Liu 2003]. However, near the free surface of the fluid, the term is usually simply discarded, which introduces an additional error in the evaluation of the gradient. We will see later on that this issue does not affect our proposed MLS approach.

The gradient SPH formula is rarely used in the form of Equation (3). It is indeed possible to derive other formulations, which are usually preferred. From $\nabla u = \nabla u \pm u\nabla 1$ we derive:

$$< \nabla u(x_i) > = \sum_i (u_i \pm u_j) V_i \nabla W_{ij}\,, \tag{4}$$

whereas from $\nabla u = (\nabla(\rho u) - u\nabla\rho)/\rho$, we get:

$$< \nabla u(x_j) > = \frac{1}{\rho_j}\sum_i (u_i - u_j) m_i \nabla W_{ij} \tag{5}$$

Finally, we can consider $(\nabla u) = \rho(\nabla(u/\rho) + (u/\rho^2)\nabla\rho)$, to obtain:

$$< \nabla u(x_j) > = \rho_j \sum_i \left(\frac{u_i}{\rho_i^2} + \frac{u_j}{\rho_j^2}\right) m_i \nabla W_{ij}\,. \tag{6}$$

The preference of one expression over another is usually driven by considerations with regard to stability or conservation. For example, Equation (4) (with the plus sign) and Equation (6) ensure that the influence of particle j over particle i is opposite to the influence of particle i over particle j. On the other hand, the forms of Equation (4) (with a minus sign) and Equation (5) guarantee that the gradient of a constant function evaluates to zero; the latter expression is computationally more efficient, but the former is more stable in the case of large differences in the density values between particles, such as in the case of multiple fluids. Further details about the 'golden rules' of SPH can be found in Monaghan [1992], for example.

2. Moving least-squares in smoothed particles hydrodynamics

The low consistency order of SPH and the errors introduced by the surface term in gradient computations can lead to highly irregular field values during the evolution of a system. This can create spurious shocks, isolated particles, potential wells, streaming and other similar artifacts. This is usually compensated for by post-processing the field values, either before or during the integration step. This can be achieved with techniques such as periodic reinitializations of the pressure or density fields (see, for example, Colagrossi and Landrini [2003]) and the XSPH method for the velocity field [Monaghan 1992], where a weighted neighborhood velocity is used to move the particle, instead of the simple particle velocity.

Although the Shepard kernel correction can be used during pressure-field initialization, this only guarantees order zero consistency, and the resulting field smoothing is often excessive. An alternative approach is based on MLS, and it can be shown that it allows SPH to achieve first-order consistency [Belytschko et al. 1996, 2000]. This is equivalent [Belytschko et al. 1998] to correcting the kernel with an affine function $B(\mathbf{x}, \xi) = \beta_0(\mathbf{x}) + \Sigma_\alpha \beta_\alpha(\mathbf{x})(\xi - \mathbf{x})_\alpha$ where α goes from 1 to the space dimension d (i.e. the summation is extended to all of the components of the vector $\xi - \mathbf{x}$).

To derive $\beta_\alpha(\cdot)$ ($\alpha = 0,1,\ldots d$), we impose:

$$u(\mathbf{x}) = \sum_i u_i W(r_i, h) V_i B(\mathbf{x}, \mathbf{x}_i)$$

for the constant function $u = u_0(\mathbf{x}) = 1$ and for the affine functions $u = u_\alpha(\xi) = \xi_\alpha - \mathrm{x}_\alpha$ (one for each space dimension). This leads to a linear system with $d + 1$ equations in $d + 1$ unknowns, which takes the matrix form:

$$\mathrm{A}(\mathbf{x})\beta(\mathbf{x}) = (1, 0, \ldots, 0)^T$$

where $\beta(\mathbf{x}) = (\beta_0(\mathbf{x}),\ldots, \beta_d(\mathbf{x}))^T$, and:

$$\mathrm{A}(\mathbf{x}) = \sum_i W(r_i, h) V_i (1, \mathbf{x}_i - \mathbf{x}) \otimes (1, \mathbf{x}_i - \mathbf{x}). \tag{7}$$

Formally, the vector function $\beta(\mathbf{x})$ is the first column of the inverse of the symmetric matrix A. If A is singular, to obtain $\beta(\mathbf{x})$ we can compute the pseudo-inverse of A, using, for example, singular value decomposition. More details about the inversion of MLS matrices can be found in Section 5.

2.1. First derivatives and the Müller approach

For a scalar field u we can write the Taylor expansion in a neighborhood of $\mathbf{x}_j$ as

$$u(\mathbf{x}) = u(\mathbf{x}_j) + \nabla u\big|_{\mathbf{x}_j}(\mathbf{x} - \mathbf{x}_j) + \ldots \simeq u_j + \nabla u\big|_{\mathbf{x}_j}(\mathbf{x} - \mathbf{x}_j) \tag{8}$$

with the shorthand notation $u_j = u(\mathbf{x}_j)$ and truncating this at first order. Writing this for $\mathbf{x} = \mathbf{x}_i$ and comparing this with

the actual field value $u_i = u(\mathbf{x}_i)$, we can evaluate the truncation error as:

$$e_{ij} = \nabla u\,\big|_{\mathbf{x}_j}\mathbf{x}_{ij} - u_{ij}$$

where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ and $u_{ij} = u_i - u_j$. Hence, the (weighted) mean square error over a neighborhood can be evaluated as:

$$< E_j >^2 \; = \sum_i e_{ij}^2 w_{ij}$$

where w_{ij} are given weights, the values of which will be discussed later on. Assuming the components of $\nabla u\,|_{\mathbf{x}_j}$ are unknown, we can find values for them that minimize $< E_j >^2$. By setting the derivatives of $< E_j >^2$ with respect to each component of $\nabla u\,|_{\mathbf{x}_j}$ equal to zero, we obtain the linear system:

$$M_j \nabla u\,\big|_{\mathbf{x}_j} = \sum_i u_{ij}\mathbf{x}_{ij} w_{ij} \tag{9}$$

where $M_j = \Sigma_i \mathbf{x}_{ij} \otimes \mathbf{x}_{ij} w_{ij}$.

This approach can be used with a variety of possible weights, but by setting $w_{ij} = V_i W_{ij}$ for an SPH kernel W, we can create a clear relationship between the MLS approach and the SPH method. This is for example the choice that was made by Müller et al. [2004], and in this case the M_j matrix is equal to the cofactor of the a_{11} element of the reinitialization matrix A as defined in Equation (7). With this choice, the MLS gradient formula becomes:

$$\nabla u(\mathbf{x}_j) = \sum_i u_{ij} M_j^+ \mathbf{x}_{ij} V_i W_{ij}\,, \tag{10}$$

where M_j^+ is the (pseudo) inverse of M_j (see also Section 5), which shows a distinct formal similarity with Equation (4), with ∇W_{ij} replaced by:

$$\tilde{\nabla} W_{ij} = M_j^+ \mathbf{x}_{ij} V_i W_{ij}\,. \tag{11}$$

With such a choice, the term $\tilde{\nabla} W_{ij}$ can be used as a correction term for the kernel gradient in any SPH formula. This has been shown to work correctly [Narayanaswamy 2008], although this use does not have the mathematical foundation of Equation (10).

We note that the general gradient formula of Equation (9) is invariant to weight scaling, in the sense that if weights $w'_{ij} = \lambda w_{ij}$ are used (with λ as a given constant), it produces the same estimate for $\nabla u(\mathbf{x}_j)$. In particular, standard SPH kernels and Shepard-corrected kernels produce the same results as from Equation (10).

Additionally, Equation (9) returns exact results for constant functions and first-order polynomials, which is proven as follows. Let $u(\mathbf{x}) = \mathbf{k} \cdot \mathbf{x} + c$ for some constant vector $\mathbf{k}$ and constant real number c; Equation (9) then becomes:

$$M_j \nabla u\,\big|_{\mathbf{x}_j} = \sum_i (\mathbf{k} \cdot \mathbf{x}_{ij})\mathbf{x}_{ij} w_{ij}$$

and since $(\mathbf{k} \cdot \mathbf{x}_{ij})\,\mathbf{x}_{ij} = \mathbf{k} \cdot (\mathbf{x}_{ij} \otimes \mathbf{x}_{ij})$, the right-hand side is exactly $\mathbf{k} \cdot M_j$, and the solution of the system gives $\nabla u\,|_{\mathbf{x}_j} = \mathbf{k}$ (which is exactly the gradient of u) due to the symmetry of M_j. This shows that the MLS approach shown so far has first-order consistency. Moreover, the first-order consistency is preserved even close to $\partial\Omega$, as the MLS gradient formula is not affected by missing surface integral terms.

A final remark is warranted concerning the neighborhoods in MLS and SPH. As discussed in Section 1.1, the accuracy of SPH is affected by both the smoothing length h of the kernel W and the average inter-particle distance H; indeed, improving accuracy in SPH requires that both h and H tend to zero, with H going to zero faster than h. This equivalently means that the density of particles in a neighborhood must *increase* as the neighborhood grows smaller.

In contrast, the MLS accuracy is only affected by the distance between the point where gradients are to be evaluated and its farthest neighbor. When using SPH kernels as MLS weights, this effectively means that the accuracy is only affected by h, and there is no requirement for the density of the particles in a neighborhood to go up when $h \to 0$. Indeed, the compact support for kernels used as MLS weights can in general be taken much smaller than the support chosen for SPH, since the gradient will still be computed accurately as long as the point has as many nonaligned neighbors as there are components in the gradient.

For comparison, consider that in a typical application to two-dimensional fluid dynamics, the ratio h/H is such that the kernel support includes about 30 or 40 particles, whereas for an MLS gradient, three or four particles (and thus effectively a kernel with $h/3$ smoothing length) would be sufficient. In practice, to reduce the chance of a singular M_j, the kernel support for MLS can be taken such that each particle has at least 10 neighbors [Müller et al. 2004].

3. Higher-order derivatives

One of the long-standing issues in SPH is the computation of higher-order derivatives, which is required when modeling the flow of a viscous fluid such as lava. Consider for simplicity the case of an isotropic, incompressible fluid with constant viscosity. The Navier–Stokes equation for the velocity is then given by:

$$\frac{D\mathbf{v}}{Dt} = -\frac{\nabla P}{\rho} + \nu\nabla^2\mathbf{v} + \mathbf{f} \tag{12}$$

where ρ is the density, $\mathbf{v}$ the velocity, P the pressure, and ν the kinematic viscosity coefficient, and $\mathbf{f}$ collects the external forces per unit mass (typically, gravity).

To compute the Laplacian of a field with SPH, it is possible to apply the same mechanism used for the gradient, obtaining:

$$\begin{aligned}\nabla u(\mathbf{x}) &\simeq \int_\Sigma W(\mathbf{x}-\mathbf{y},h)\nabla_y u(\mathbf{y})\cdot\mathbf{n}d\Sigma + \\ &+ \int_\Omega \nabla_y u(\mathbf{y})\nabla_x W(\mathbf{x}-\mathbf{y},h)d\mathbf{y} = \\ &\simeq \int_\Sigma W(\mathbf{x}-\mathbf{y},h)\nabla_y u(\mathbf{y})\cdot\mathbf{n}d\Sigma + \\ &+ \int_\Sigma u(\mathbf{y})\nabla_x W(\mathbf{x}-\mathbf{y},h)d\mathbf{y}\cdot\mathbf{n}d\Sigma + \\ &- \int_\Omega u(\mathbf{y})\nabla_x^2 W(\mathbf{x}-\mathbf{y},h)d\mathbf{y}.\end{aligned}$$

Once again, under the additional assumption that the support of W does not intersect $\Sigma = \partial\Omega$, we would obtain the SPH interpolation:

$$<\nabla^2 u(\mathbf{x})> = -\sum_i u_i V_i \nabla^2 W(r_i,h).$$

However, the error induced by discarding the surface integrals for points near Σ is now much higher than in the gradient case, and finding correcting terms to take this into consideration is even harder.

A different approach to discretize the Laplacian was first used by Cleary and Monaghan [1999] to model thermal conduction, and it was adapted by Morris et al. [1997] for the dynamic case. This is derived from a Taylor expansion of the SPH gradient, and it takes the form:

$$<\nabla^2\mathbf{v}_j> = \sum_i m_i \frac{(\rho_i+\rho_j)\mathbf{x}_{ij}\cdot\nabla W_{ij}}{\rho_i\rho_j(\mathbf{x}_{ij}^2+\varepsilon h^2)}\mathbf{v}_{ij} \qquad (13)$$

where $\varepsilon \simeq 0.01$ and h is the smoothing length. The εh^2 term is included to prevent singularities in the case of overlapping particles. In the case of symmetric (non-corrected) kernels, Equation (1) can be used to simplify Equation (13) to:

$$<\nabla^2\mathbf{v}_j> = \sum_i m_i \frac{(\rho_i+\rho_j)\mathbf{v}_{ij}}{\rho_i\rho_j}F_{ij} \qquad (14)$$

where $F_{ij} = F(r_{ij}, h)$ with F defined as in Equation (2). This expression has the benefit of being computable even in the case of overlapping particles, provided that F can be computed analytically.

3.1. Moving least-squares for second-order derivatives

It is possible to use the first-order MLS gradient described in Section 2.1 as a kernel correction, replacing the kernel gradient in Equation (13) with the corrected kernel gradient of Equation (11). Since we have:

$$\mathbf{x}_{ij}\cdot\tilde{\nabla}W_{ij} = \mathbf{x}_{ij}M_j^+\mathbf{x}_{ij}W_{ij} = \mathbf{x}_{ij}^2\hat{d}_{ij}M_j^+\hat{d}_{ij}W_{ij}$$

where $\hat{d}_{ij}$ is the unit vector (direction) of $\mathbf{x}_{ij}$, this allows us to simplify Equation (13) to:

$$<\nabla^2\mathbf{v}_j> = \sum_i m_i \frac{(\rho_i+\rho_j)\mathbf{v}_{ij}}{\rho_i\rho_j}\tilde{F}_{ij} \qquad (15)$$

where $\tilde{F}_{ij} = \hat{d}_{ij}M_j^+\hat{d}_{ij}W_{ij}$.

However, this approach does not take full advantage of the power of MLS, and suffers from all of the drawbacks of the SPH method. Instead, we can extend the first-order MLS method from Section 2.1 to allow direct computation of higher derivatives. For simplicity, we illustrate the approach in the two-dimensional case. The vector components will be (x,y) and the derivatives will be written as $u_{,x}$ for $\partial u/\partial x$, and so forth for the other derivatives. The Taylor expansion for u around $\mathbf{x}_j$ can then be written up to the second error:

$$\begin{aligned}u(\mathbf{x}) = u(\mathbf{x}_j) &+ u_{,x}(x-x_j) + u_{,y}(y-y_j) + \\ &+ u_{,xx}\frac{(x-x_j)^2}{2} + u_{,xy}(x-x_j)(y-y_j) + \\ &+ u_{,yy}\frac{(y-y_j)^2}{2} + \ldots\end{aligned}$$

and the truncation error when evaluating the expression for $\mathbf{x} = \mathbf{x}_i$ is now:

$$e_{ij} = u_{,x}x_{ij} + u_{,y}y_{ij} + u_{,xx}\frac{x_{ij}^2}{2} + u_{,xy}x_{ij}y_{ij} + u_{,yy}\frac{y_{ij}^2}{2} - u_{ij}\ .$$

Once again, we want to minimize the mean square error $<E_j>^2 = \sum_i e_{ij}^2 w_{ij}$ with respect to the partial derivatives. For simplicity, we can write:

$$\begin{aligned}\bar{\mathbf{x}}_{ij} &= \left(x_{ij}, y_{ij}, \frac{x_{ij}^2}{2}, x_{ij}y_{ij}, \frac{y_{ij}^2}{2}\right) \\ \bar{\nabla} &= \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial x^2}, \frac{\partial}{\partial x\partial y}, \frac{\partial}{\partial y^2}\right).\end{aligned}$$

Then equating the derivatives of $<E_j>^2$ to zero with respect to the components of $\bar{\nabla}u$ gives us the linear system:

$$\bar{M}_j\bar{\nabla}u = \sum_i u_{ij}\bar{\mathbf{x}}_{ij}w_{ij} \qquad (16)$$

where $\bar{M}_j = \sum_i \bar{\mathbf{x}}_{ij}\otimes\bar{\mathbf{x}}_{ij}w_{ij}$. The new formulae are formally identical to the first-order formulae, with an "extended" relative position vector $\bar{\mathbf{x}}_{ij}$ that includes higher powers and an "extended" differential operator $\bar{\nabla}$ that includes higher-order derivatives.

A similar approach to the one described here can be found with the finite pointset method (FPM) that was developed by Tiwari and Kuhnert [2001]. In the FPM, second-order MLS are used to locally and iteratively solve the Poisson equation for the pressure in an incompressible fluid. In this case, it is also necessary to include boundary conditions in Equation (16), both on the free surface and on the physical boundaries of the fluid. In particular, this requires the detection of the surface particles of the fluid, which loses one of the main benefits of particle methods such as SPH, which is the automatic and implicit tracking of surfaces.

Instead, we propose to use second-order MLS in an approach that fits better with the standard SPH modeling of fluid dynamics. In this respect, our approach follows the method described by Dilts [1999, 2000] more closely, with interpolants that are centered on the particle for which the derivatives are being computed. As noted in Belytschko et al. [1998], this improves the accuracy and stability of the computations, and brings our approach within the general framework described in Schrader et al. [2010].

The use of MLS over the standard SPH discretization gives us higher accuracy for first-order derivatives, as well as the possibility to directly compute second-order derivatives. For example, the Laplacian in two dimensions can be evaluated from Equation (16) with:

$$\nabla^2 = \sum_i u_{ij} \left(\bar{M}_j^+ \bar{\mathbf{x}}_{ij} \right)_{xx+yy} w_{ij} \tag{17}$$

where $\left(\bar{M}_j^+ \bar{\mathbf{x}}_{ij} \right)_{xx+yy}$ denotes the sum of the third and fifth components of the $\bar{M}_j^+ \bar{\mathbf{x}}_{ij}$ vector, i.e. the components corresponding to the pure derivatives of order two. A similar formula applies with more than two dimensions.

For second-order MLS, we can make considerations similar to those for first-order MLS. For example, with the appropriate choice of the weighting functions, we can use MLS as a gradient correction for SPH, effectively replacing ∇W_{ij} in any SPH equation with the first components of:

$$\tilde{\tilde{\nabla}} W_{ij} = \bar{M}_j^+ \bar{\mathbf{x}}_{ij} W_{ij}. \tag{18}$$

Second-order MLS shows second-order consistency (the proof follows that for first-order consistency of first-order MLS), and it is not affected by the surface integral terms that affect SPH derivatives of any order, so that the accuracy is preserved even near the boundary of Ω, as long as there are enough neighboring particles. The minimum number of particles is now higher, with five neighbors as the bare minimum for the computation of all of the derivatives in two dimensions. Additionally, the particles must not be laid out on a line (or in a plane in three dimensions), like in the first-order case, nor in a cross shape, as otherwise this might prevent mixed derivatives from being computable. In general, the neighborhoods for second-order MLS must be larger than those for first order MLS, although the number of neighbors in the compact support of an SPH kernel is still sufficient.

4. An example

To illustrate the benefits of our approach, we compare the results for a classic two-dimensional dam-break problem, where a mass of fluid with an initial rectangular configuration $H_0 = 2$ m high and 1 m wide is left free to flow within a 4 m × 4 m box.

The fluid is assumed to be isotropic and quasi-compressible. Its motion is driven by the Navier–Stokes equations paired with an equation of state:

$$\frac{D\mathbf{v}}{Dt} = -\frac{\nabla P}{\rho} + \nu \nabla^2 \mathbf{v} + \mathbf{g} \tag{19}$$

$$\frac{D\rho}{Dt} = -\rho \nabla \cdot \mathbf{v}, \tag{20}$$

$$P = \frac{\rho_0 c_s^2}{\gamma} \left(\left(\frac{\rho}{\rho_0} \right)^\gamma - 1 \right),$$

where $\mathbf{g} = (0, -g)$ is gravity ($g = 9.81$ m$^2/s$), $\rho_0 = 1{,}000$ kg/m^3 is the density at rest, $\gamma = 7$, and the speed of sound c_s is chosen to be an order of magnitude higher than the maximum speed expected during the flow, which in our problem is $\sqrt{2gH_0} \sim 6.3$ m/s.

Boundary conditions are implemented using an approach similar to the one described by Liu et al. [2002]. Near a physical boundary, a particle interacts not only with its own neighbors, but also with mirror images of itself and of all of its neighbors. These ghost particles have material properties that match those of their original, except for the velocity.

If no-slip conditions are wanted at the border, the mirror velocities are opposite to their original ones, whereas free-slip conditions are implemented by giving the ghost particles a mirror velocity, where only the velocity component normal to the boundary is rejected. Additional ghost particles with opposite velocities are also added at corners where two linear components of the containing box meet. As we want to simulate a viscous fluid, we use no-slip conditions in this example.

As ghost particles are not sufficient to guarantee that the boundary will not be penetrated [Liu et al. 2002], particles that exert a short-range repulsive Lennard–Jones force are also added. In contrast to Liu et al. (2002), however, we do not establish these particles *a priori*, but they are generated at run-time as a projection of the particle itself on the boundary.

This strategy is particularly efficient in our case, because the Lennard–Jones particle position can be found quickly as the midpoint of the original particle position and its ghost, which reduces the number of particles that need to be tracked during computation. Additionally, the force exerted by these virtual Lennard–Jones particles is constant for a particle traveling parallel to a border, which solves one of the famous issues with Lennard–Jones boundary particles [Monaghan and Kajtar 2009].

Our choice for a smoothing kernel both in SPH and as weighting function in MLS, is based on the $\psi_{3,1}$ function by Wendland [1995]. We have $W(r,h) = w(r/h)/h^2$, where:

$$w(q) = \begin{cases} \frac{7}{4\pi} \left(1 - \frac{q}{2} \right)^4 (2q + 1) & q \le 2 \\ 0 & q > 2 \end{cases},$$

We can also define F from Equation (2) as $F(r,h) = f(r/h)/h^4$,

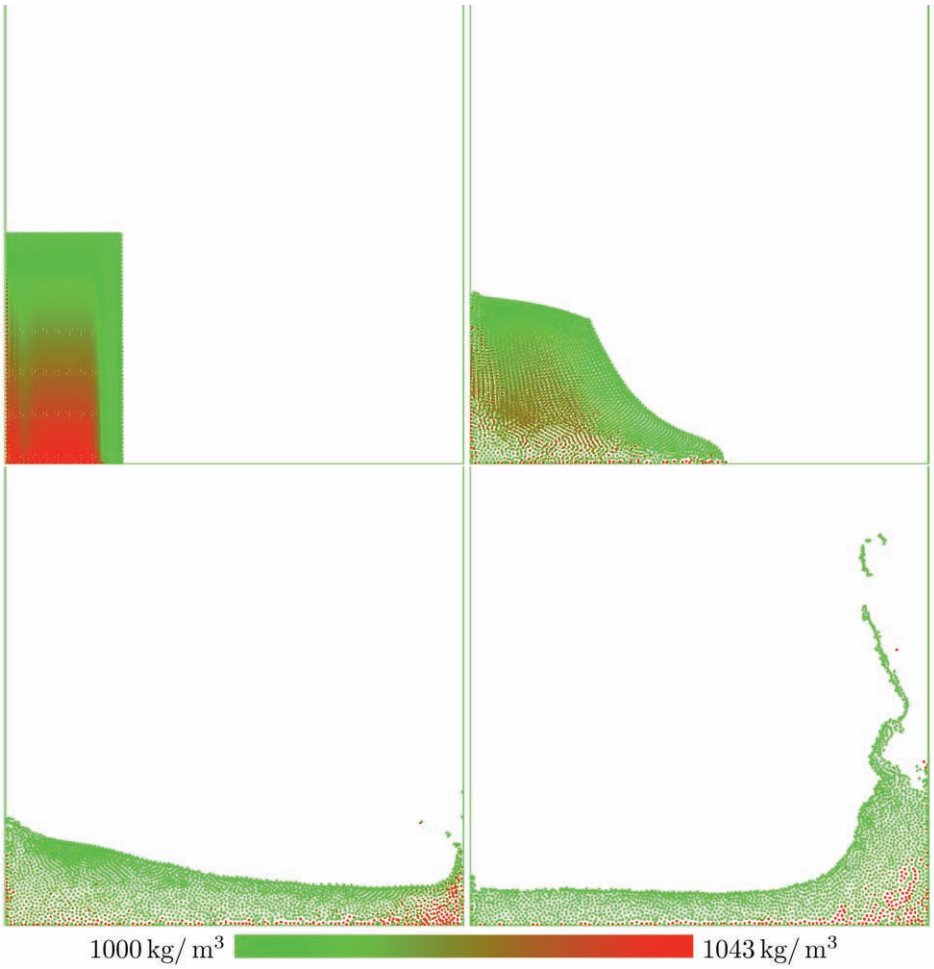

Figure 1. Dam-break two-dimensional evolution, at $t = 0.01\ s$; $0.4\ s$; $0.8\ s$; $1.6\ s$, using standard SPH. Color coding by density.

where:

$$f(q) = \begin{cases} \frac{35}{32\pi}(q-2)^3 & q \le 2 \\ 0 & q > 2 \end{cases},$$

so that $\nabla W(\mathbf{x}_{ij}, h) = \mathbf{x}_{ij} F(r_{ij}, h)$.

The fluid has a kinematic viscosity $\nu = 10^{-6}\,\mathrm{m}^2/\mathrm{s}$ (water), so we apply the viscosity formula of Equation (14). The discretized equations of motion can then be written as:

$$\frac{D\rho_j}{Dt} = \sum_i \frac{\rho_j}{\rho_i} \mathbf{v}_{ij} \cdot \mathbf{x}_{ij} F(r_{ij}, h) m_i \ ,$$

$$\frac{D\mathbf{v}}{Dt} = -\sum_i \left(\frac{P_i + P_j}{\rho_i \rho_j} + \Pi_{ij} \right) \mathbf{x}_{ij} F(r_{ij}, h) m_i + \\ + \nu \sum_i \frac{m_i (\rho_i + \rho_j) \mathbf{v}_{ij}}{\rho_i \rho_j} F_{ij}$$

where Π_{ij} is an artificial viscosity that takes the form:

$$\Pi_{ij} \begin{cases} -\alpha \frac{\bar{c}_s}{\bar{\rho}} h \frac{\mathbf{x}_{ij} \cdot \mathbf{v}_{ij}}{r_{ij} + \varepsilon h^2} & \mathbf{x}_{ij} \cdot \mathbf{v}_{ij} < 0, \\ 0 & \text{otherwise} \end{cases}$$

where $\bar{c}_s$ is the average speed of sound computed at the particles i and j, and $\bar{\rho}$ their average density. The artificial viscosity coefficient is $\alpha = 0.2$ in our case, and the correcting factor εh^2 (with $\varepsilon = 0.01$) prevents singularities in the case of overlapping particles. More details on the artificial viscosity in SPH can be found in Monaghan and Gingold [1983].

In this test, we compare with second-order MLS used as a gradient correction, with $\mathbf{x}_{ij} F_{ij}$ replaced by $\tilde{\nabla} W_{ij}$ from Equation (18), except for the viscous term that relies on Equation (17) to compute the Laplacian of the velocity.

A second comparison is carried out with second-order MLS derived from the differential Equations (19) and (20); in this case, the discrete formulation becomes:

$$\frac{D\mathbf{v}_i}{Dt} = -\frac{\nabla P_i}{\rho_i} + \tilde{\nu} \frac{\nabla^2 \mathbf{v}}{\rho_i} + \mathbf{g} \ ,$$

$$\frac{D\rho_j}{Dt} = -\rho_i \nabla \cdot \mathbf{v}_i \ ,$$

with Equations (16) and (17) to compute the gradients, divergences and Laplacians. The viscosity $\tilde{\nu}$ is obtained by adding $\alpha c_s h$ to ν when $\nabla \cdot \mathbf{v}_i < 0$. In this case, Cholesky decomposition is used to solve the linear systems, as detailed in Section 5.

In all three cases, integration is carried out using an explicit predictor/corrector scheme described later, with a dynamic

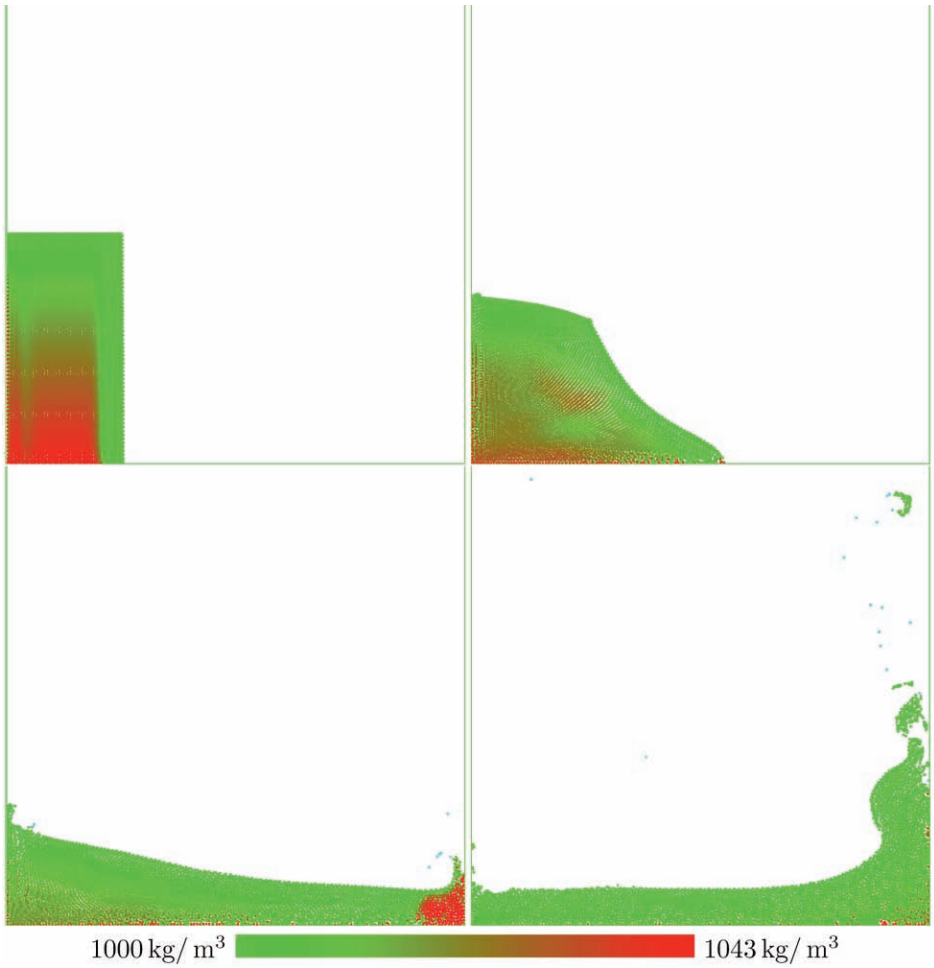

Figure 2. Dam-break two-dimensional evolution, at $t = 0.01$ *s*; 0.4 *s*; 0.8 *s*; 1.6 *s*, using second-order MLS as a kernel correction in SPH. Color coding by density.

time-step that is limited by standard Courant-Friedrichs-Lewy conditions:

$$\Delta t \leq k \ \min\left\{\frac{h}{c_s}, \sqrt{\frac{h}{f_M}}, \frac{h^2}{\nu}\right\}$$

where f_M denotes the maximum particle force per unit mass, and we take $k = 0.3$. With $h = 0.02$, this gives us $\Delta t \sim 10^{-4}$.

From the particle positions X_n and velocities V_n at time t_n, we compute the particle accelerations $F^*_{n+1}(X_n, V_n)$ and the maximum timestep Δt^*; the predicted new positions and velocities are then given by:

$$V^*_{n+1} = V_n + \Delta t^* F^*_{n+1} ,$$
$$X^*_{n+1} = X_n + \Delta t^* V^*_{n+1} .$$

The correction step evaluates $F^{**}_{n+1}\left(X^*_{n+1}, V^*_{n+1}\right)$ and a new maximum time-step Δt^{**}. We then compute the maximum time-step as $\Delta t = \min(\Delta t^*, \Delta t^{**})$, and the accelerations as $F_{n+1} = \left(F^*_{n+1} + F^{**}_{n+1}\right)/2$, and integrate with:

$$V_{n+1} = V_n + \Delta t \ F_{n+1} ,$$
$$X_{n+1} = X_n + \Delta t \ V_{n+1} .$$

The evolution of the dam break with SPH (Figure 1) shows typical issues in SPH-driven simulations, with a noisy density field that leads to streaks of high-density particles that infiltrate the main body flow. In addition, the absence of correction terms for the tensile instability [Monaghan 2000] leads to particle overlapping in the lower area of the flow, with a consequent reduction in precision and an apparent rarefaction of the particle positions.

In contrast, the MLS-driven simulations (Figures 2 and 3) show much cleaner behavior, with the density field evolving smoothly without any need for post-processing phases or reinitialization. Some noise is still present, particularly in the region closer to the physical boundary; this effect does not have a strong influence on the main body flow, although it allows a few particles to detach when the fluid layer is very thin.

We can also compare our results with the experimental timing obtained by Martin and Moyce [1952], which match our case with $a = 1$ m and $n^2 = 2$. For the wavefront Z, the timing is scaled by $n\sqrt{g/a} \sim 4.43$ s and the wavefront Z is scaled by a; the results are shown in Table 1. The column height H is expressed as a fraction of the original column height H_0, and the timing is scaled by $n\sqrt{g/a} \sim 3.13$ s; these results are shown in Table 2.

We observe that MLS gave the same timing (within two digits of accuracy) used both as a kernel correction and directly, and the results tend to be closer to the experimental

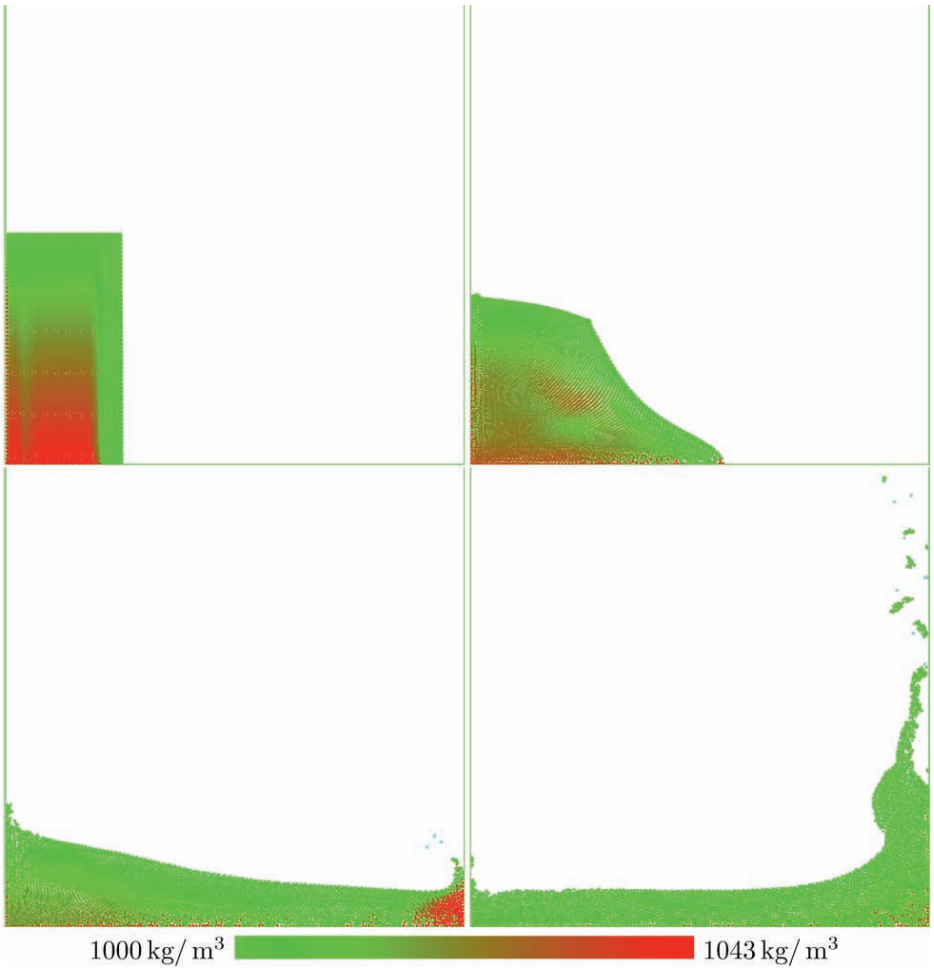

Figure 3. Dam-break two-dimensional evolution at $t = 0.01\ s$; $0.4\ s$; $0.8\ s$; $1.6\ s$, using second-order MLS from the analytical formulation. Color coding by density.

Z	T range (exp)	T (SPH)	T (MLS-corr)	T (MLS)
1.11	0.30 − 0.48	0.40	0.40	0.40
1.89	1.56 − 1.67	1.46	1.51	1.51
2.33	1.87 − 2.02	1.90	1.92	1.92
2.78	2.21 − 2.38	2.31	2.31	2.31

Table 1. Time to reach wavefront distance Z in experiments as compared to timing achieved with SPH, second-order MLS as a kernel correction, and MLS from the analytical formulation. Scales as detailed in the text.

H	T range (exp)	T (SPH)	T (MLS-corr)	T (MLS)
0.89	0.76 − 0.79	0.75	0.79	0.79
0.78	1.05 − 1.11	1.10	1.07	1.07
0.72	1.28 − 1.33	1.32	1.29	1.29
0.67	1.44 − 1.47	1.48	1.54	1.54
0.61	1.66 − 1.68	1.73	1.82	1.82

Table 2. Time at which column height H reaches a given fraction of the original height in experiments as compared to timing achieved with SPH, second-order MLS as a kernel correction, and MLS from the analytical formulation. Scales as detailed in the text.

values than SPH. The major exception is for the lower values of H, for which both particle methods, and MLS in particular, tend to lag behind the experimental data. This appears to be related to the observable small peak adherent to the left boundary and clearly visible in Figures 2 and 3, an artifact of boundary conditions that appears to be exerting excessive drag in this case. If this issue is taken into account, the corrected timing for MLS fall back within the experimental range.

5. Implementation notes

The MLS matrices are symmetric and positive semi-definite. Rather than computing the inverse, it is therefore generally more efficient to compute the Cholesky decomposition in the non-radical form; i.e. to find a lower triangular matrix L with unitary diagonal, and a diagonal matrix D such that $M = LDL^T$, and then to solve the linear systems associated with any derivatives we need to compute, rather than using the matrix dot vector notation synthetically used throughout this report.

Such an approach, however, is only possible when the matrix is strictly definitely positive, as otherwise the standard constructive Cholesky decompositions fail when elements of D are null or very close to zero. In such a case, a better approach is to compute a pseudo-inverse of M and then using

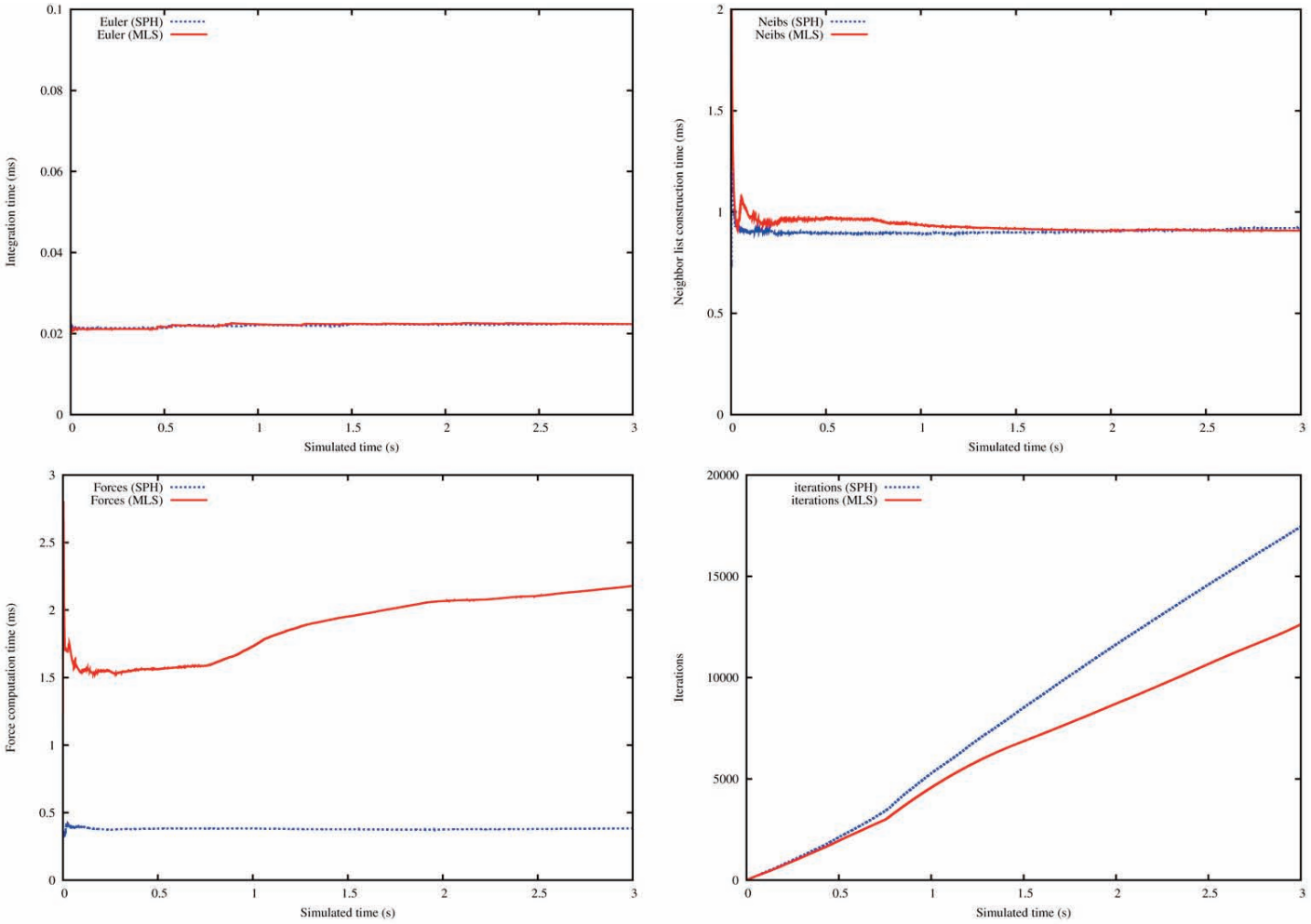

Figure 4. Code performance for SPH versus MLS implementation of the dam break. The average Euler integration, neighbor list construction, forces computation times, and number of iterations are plotted against the total simulated time.

the formulae detailed in this report in their actual forms.

Any symmetric, positive, semi-definite matrix can be written [Press et al. 1992, Golub and Van Loan 1996] in the form $M = G^T E G$, where G is an orthogonal matrix ($G^T G = G G^T = I$) and E is a diagonal matrix where its diagonal elements are the singular values of M. To find G and E, we use the iterative Jacobi method [Golub 2000], which writes G as a product of Givens rotations.

The pseudo-inverse E^+ of E is then computed as the diagonal matrix:

$$E^+_{ii} = \begin{cases} 1/E_{ii} & E_{ii} \neq 0 \\ 0 & E_{ii} = 0 \end{cases},$$

and the pseudo-inverse of M is obtained as $M^+ = G^T E^+ G$.

In practice, both during the Cholesky decomposition and for the pseudo-inverse creation, the zero test is replaced by $|A_{ij}| < \varepsilon$ where A_{ij} is the matrix element to be tested and ε is an appropriately small number.

Picking too small an ε can lead to disproportionately large reciprocals in M^+, with a consequent excess force being exerted on the corresponding particles, which can result in large instabilities during a simulation; on the other hand, too large an ε will increase the numerical error in the pseudo-inverse computation, with a consequent perturbation of the particle motion.

Our implementation, which is based on the work by Hérault et al. [2010a], exploits the parallel computing power offered by NVIDIA graphics processing units (GPUs) to achieve two orders of magnitude in speed-up over the equivalent code for standard central processing units (CPUs). However, it is also limited by the single precision native to the hardware by the GPUs themselves. In this case, we found that the best value for ε lies in the range $d_M \varepsilon_M / 2 \leq \varepsilon \leq 2 d_M \varepsilon_M$ where d_M is the dimension of the MLS matrix (e.g. $d_M = 5$ for a second-order MLS with two space dimensions) and ε_M is the machine epsilon for single precision.

5.1. Performance

With respect to standard SPH, the MLS approach has additional computational cost that is associated with the generation and use of the MLS matrix for each particle. The impact of the extra computations on our GPU implementation running on a GTX 280 is shown in Figure 4. While the average execution time of the integration (Euler)

and the neighbor list construction (Neibs) kernels are essentially the same with both SPH and MLS, the force computation (Forces kernel) is relatively consistently about 4 times greater.

The additional cost per iteration is off-set by a slightly larger time-step in MLS that leads to a lower number of total iterations to cover the simulated time. Although this result might appear a little surprising, it is easily explained by the highly irregular fields that cause spuriously large forces in SPH. As a consequence, the total run-time for MLS (158 *s*) is only about 60% longer than for pure SPH (98 *s*).

6. Conclusions

MLS are typically used in SPH in the form of either a post-processing smoothing filter for the density, or a kernel correction for gradient evaluation. In both cases, their use is based on a first-order Taylor series expansion. We propose a method to extend the kernel correction approach by using second-order MLS, which provides improved approximations for gradients as well as a robust formula for second-order derivatives.

The new approach can be used both as a kernel correction in SPH, which preserves the classic SPH formalism and replacing the ∇W terms with a matrix/vector product, and also directly, using discrete equations that are specific to MLS.

The application of the method described to the solution of the Navier–Stokes equations for a viscous fluid shows visible benefits over the use of standard SPH, which reduces the need for post-processing filters, such as XSPH, Shepard or MLS itself. The computational cost of our method is, however, about 60% greater than that of standard SPH.

A more thorough analysis of the influence of the weighting functions chosen during the MLS process remains to be undertaken, together with an analysis of the opportunity to choose different forms for the derivative computation by taking advantage of the standard gradient identities from which the distinct SPH gradient formulae are derived. Such an analysis should identify the most appropriate form to guarantee that the discrete system will preserve quantities such as total momentum and angular momentum when they are conserved at the analytical level.

Acknowledgements. This study was undertaken with financial support from the V3-LAVA project (INGV-DPC 2007-2009 contract).

References

Belytschko, T., Y. Krongauz, D. Organ, M. Fleming and P. Krysl (1996). Meshless methods: an overview and recent developments, Comput. Method. Appl. M., 139, 3-47.

Belytschko, T., Y. Krongauz, J. Dolbow and C. Gerlach (1998). On the completeness of meshfree particle methods, Int. J. Numer. Meth. Eng., 43, 785-819.

Belytschko, T., Y. Guo, W. Kam Liu and S. Ping Xiao (2000). A unified stability analysis of meshless particle methods, Int. J. Numer. Meth. Eng., 48 (9), 1359-1400.

Cleary, P.W. and J.J. Monaghan (1999). Conduction modelling using smoothed particle hydrodynamics, J. Comput. Phys., 148, 227-264.

Colagrossi, A. and M. Landrini (2003). Numerical simulation of interfacial flows by smoothed particle hydrodynamics, J. Comput. Phys., 191, 448-475.

Dalrymple, R.A. and A. Hérault (2009). Levee Breaching with GPU-SPHysics Code, In: Proceedings of the 4th international SPHERIC workshop (Nantes, France, May 2009), available at: http://www.ce.jhu.edu/dalrymple/GPU/SPHERICIVDalrympleHerault.pdf.

Dilts, G.A. (1999). Moving-least-squares-particle hydrodynamics I. Consistency and stability, Int. J. Numer. Meth. Eng., 44 (8), 1115-1155.

Dilts, G.A. (2000). Moving least-squares particle hydrodynamics II: conservation and boundaries, Int. J. Numer. Meth. Eng., 48 (10), 1503-1524.

Gingold, R. and J. Monaghan (1977). Smoothed particle hydrodynamics – Theory and application to non-spherical stars, Mon. Not. R. Astron. Soc., 181, 375-389.

Golub, G.H. and C.F. Van Loan (1996). Matrix Computations, Johns Hopkins Studies in Mathematical Sciences, Johns Hopkins University Press, 728 pp.

Golub, G. (2000). Eigenvalue computation in the 20th century, J. Comput. Appl. Math., 123 (1-2), 35-65.

Hérault, A., A. Vicari, C. Del Negro and R.A. Dalrymple (2009). Modeling Water Waves in the Surf Zone with GPU-SPHysics, In: Proceedings of the 4th international SPHERIC workshop (Nantes, France, May 2009), available at: http://www.ce.jhu.edu/dalrymple/GPU/SphericHerault.pdf.

Hérault, A., G. Bilotta and R.A. Dalrymple (2010a). SPH on GPU with CUDA, J. Hydraul. Res., 48 (extra issue), 74-79.

Hérault, A., G. Bilotta, C. Del Negro, G. Russo and A. Vicari (2010b). SPH modeling of lava flows with GPU implementation, In: L. Fortuna, A. Fradokv and M. Frasca (eds.), From physics to control through an emergent view, World Scientific Series on Nonlinear Science, Series B, 15, 183-188.

Hérault, A., G. Bilotta, A. Vicari, E. Rustico and C. Del Negro (2011). Numerical simulation of lava flow using a GPU SPH model, Annals of Geophysics, 54 (5), 600-620 (this issue).

Lanson, N. and J.P. Vila (2008). Renormalized Meshfree Schemes I: Consistency, Stability, and Hybrid Methods for Conservation Laws, SIAM J. Numer. Anal., 46 (4), 1912-1934.

Liu, M., G. Liu and K. Lam (2002). Investigations into water mitigation using a meshless particle method, Shock Waves, 12 (3), 181-195.

Liu, G. and M. Liu (2003). Smoothed Particle Hydrodynam-

ics: A Meshfree Particle Method, World Scientific Publishing Company, 449 pp.
Lucy, L. (1977). A numerical approach to the testing of the fission hypothesis, Astron. J., 82, 1013-1024.
Martin, J. and W. Moyce (1952). Part IV. An experimental study of the collapse of liquid columns on a rigid horizontal plane, Philos. T. R. Soc. A, 244 (882), 312-324.
Monaghan, J. and R. Gingold (1983). Shock simulation by the particle method SPH, J. Comput. Phys., 52 (2), 374-389.
Monaghan, J. (1992). Smoothed Particle Hydrodynamics, Annu. Rev. Astron. Astr., 30 (1), 543-574.
Monaghan, J. (2000). SPH without a Tensile Instability, J. Comput. Phys., 159 (2), 290-311.
Monaghan, J. and J. Kajtar (2009). SPH particle boundary forces for arbitrary boundaries, Comput. Phys. Commun., 180 (10), 1811-1820.
Morris, J.P., P.J. Fox and Y. Zhu (1997). Modeling Low Reynolds Number Incompressible Flows Using SPH, J. Comput. Phys., 136 (1), 214-226.
Müller, M., R. Keiser, A. Nealen, M. Pauly, M. Gross and M. Alexa (2004). Point based animation of elastic, plastic and melting objects, In: R. Boulic and D.K. Pai (eds.), SCA '04 Symposium on Computer Animation 2004 (Grenoble, France, 2004), Eurographics Association Aire-la-Ville, Switzerland, 141-151.
Narayanaswamy, M.S. (2008). A hybrid Boussinesq-SPH wave propagation model with applications to forced waves in rectangular tanks, Ph.D. thesis, Johns Hopkins University.
Press, W.H., S.A. Teukolsky, W.T. Vetterling and B.P. Flannery (1992). Numerical Recipes in C: The Art of Scientific Computing, 2nd edition, New York, 994 pp.
Schrader, B., S. Reboux and I.F. Sbalzarini (2010). Discretization correction of general integral pse operators for particle methods, J. Comput. Phys., 229, 4159-4182; URL: http://dx.doi.org/10.1016/j.jcp.2010.02.004.
Tiwari, S. and J. Kuhnert (2001). Finite pointset method based on the projection method for simulations of the incompressible Navier-Stokes equations, Berichte des Fraunhofer ITWM, 30, 1-18; available at: http://www.itwm.fraunhofer.de/fileadmin/ITWM-Media/Zentral/Pdf/Berichte_ITWM/2001/bericht30.pdf.
Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree, Adv. Comput. Math., 4 (1), 389-396.

*Corresponding author: Giovanni Russo,
Università di Catania, Dipartimento di Matematica e Informatica,
Catania, Italy; email: russo@dmi.unict.it.