Particle Methods and Waves, with Emphasis on SPH

Robert A. Dalrymple

Department of Civil Engineering
Johns Hopkins University

June, 2007

Particle Methods for Water Waves

Robert A. Dalrymple
Johns Hopkins University

Abstract

The purpose of this section is to examine the numerical model of water waves, particularly with a method of meshless methods, such as Smoothed Particle Hydrodynamics (SPH). This part of the course will deal with numerical models, numerical interpolation, and meshfree method, with application to SPH for waves.

1 Numerical Modeling

The study of water waves has taken the usual scientific paths–experimentation, laboratory studies, theoretical work, and now numerical methods.

One of the earliest theoretical models was the linear wave theory due to Airy (1845).[1] This model (described completely in Dean and Dalrymple (1991)) assumed that the water was incompressible and the flow irrotational, meaning that the viscosity of the fluid was neglected. These assumptions allowed the use of a velocity potential, $\phi(\mathbf{x}, t)$, from which wave-induced velocities could be obtained via $-\nabla\phi = \mathbf{u}$ (note the use of the arbitrary minus sign.) The equation for the velocity potential is the *Laplace* equation:

$$\nabla^2\phi(\mathbf{x}, t) = 0 \tag{1}$$

Associated with this governing equation for the fluid domain are boundary conditions that exist at the bottom, the free surface, and the lateral sides of the domain. The domain will be taken as a rectangle, with the coordinate system consisting of x and y in the horizontal plane and z in the vertical. The rectangle's base is the ocean bottom at $z = -h(x, y)$ and a free surface, denoted by $z = \eta(x, y, t)$, is the top.

At the bottom, the flow is usually taken as zero through the boundary, or $-\nabla\phi \cdot \mathbf{n} = 0$ on the bottom, $z = -h(x, y)$. The free surface, which moves in response to pressure in the fluid, requires two boundary conditions; one to locate the boundary and one to prescribe constant pressure. These are the Kinematic Free Surface Boundary Condition (KFSBC) and that Dynamic Free Surface Boundary Conditions (DFSBC).

A variety of analytical models have been developed for waves of permanent form propagating over horizontal bottoms. The linear Airy theory, which provided the dispersion relationship relating the wave period, the wave length, and the water depth.

$$\omega^2 = gk \tanh kh$$

where the angular frequency $\omega = 2\pi/T$, where T is the wave period, and the wave number $k = 2\pi/L$. This equation is a transcendental equation and has to be iteratively.

Stokes (1847) extended the Airy theory to include nonlinear terms. The Stokes wave theory is still used in offshore design. In shallow water, the Stokes theory breaks down, and the theory of Boussinesq (1872) is preferred. There has been a tremendous resurgence of this theory recently due to the ability to extend the model from shallow water into deeper water.

The advent of computers allowed the extension of Stokes-type models to very high order Dean (1965)

[1]Craik (2004) presents an early history of water wave theory. He indicates that Airy's chapter was likely published in 1841.

2 Numerical Interpolation

Given some data points, how do we fit a curve through the data in the best way? Alternatively, if we are given certain points and each point has an associated value, can we approximate the mathematical function that produced the values? There are many ways available for fitting data that is one-dimensional, such as polynomial fits and least squares fits. Polynomial fits are attractive, since the Weierstrass approximation theorem, which says any continuous function can be fit by a polynomial in the range of $a \leq x \leq b$. The fitting polynomial passes through all the data points exactly.

2.1 Polynomial Interpolant

For example, there may be a function, $y(x)$ for which we know four points: $(x_i, y_i), i = 1, 4$. Let's fit a polynomial to these points:

$$y_p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

There are exactly four unknowns in the polynomial, the a_i values. (The number of a values must be equal to the number of collocation points.) If we force the polynomial to pass through the given y value for each of the four x values (using a method called *collocation*), then we have, in matrix form,

$$\begin{pmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \\ 1 & x_4 & x_4^2 & x_4^3 \end{pmatrix} \begin{pmatrix} a_o \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \tag{2}$$

or, we can write this more compactly as

$$\mathbf{P}\,\mathbf{a} = \mathbf{y} \tag{3}$$

The solution for the a_i values is

$$\mathbf{a} = \mathbf{P}^{-1}\,\mathbf{y} \tag{4}$$

Example 1

Consider the following four pairs of points: (0,2), (1,0.3975), (2, -0.1126), (3, -0.0986). These points are plotted in Figure 1 as blue dots. Substituting these values into the equation above, we solve for the a values by matrix inversion. The fitting polynomial is then $y_p(x) = 2.0 - 2.3380x + 0.8302x^2 - 0.0947x^3$. This polynomial is plotted in red in the figure along with the data points and the exact function in blue–$y(x) = 2\exp(-x)\cos(x)$. Note that $y_p(x)$ goes through the data as desired. There is a pretty good match in the range $0 \leq x \leq 3$.

As a caveat, notice what happens when you use the polynomial fit outside the range of fitting as shown in Figure 2. This is because a cubic polynomial has three zero crossings and several were in the range shown.

In principle, we can fit exactly $N + 1$ points with an N order polynomial. However, higher order polynomial fits, while intuitively more accurate, have more zero crossing. (Further the matrix on the left side of Eqn. 3 becomes poorly conditioned.) This usually proves to be a problem with more than a handful of points, so other methods, such as cubic splines, which fits different cubic polynomials between each contiguous pair of points. Neighboring polynomials are matched at the common data points as are their first two derivatives.

In the example above, the data were equally spaced along the x axis. The method also works with unequally spaced data. The method can also be applied in more than one dimension.

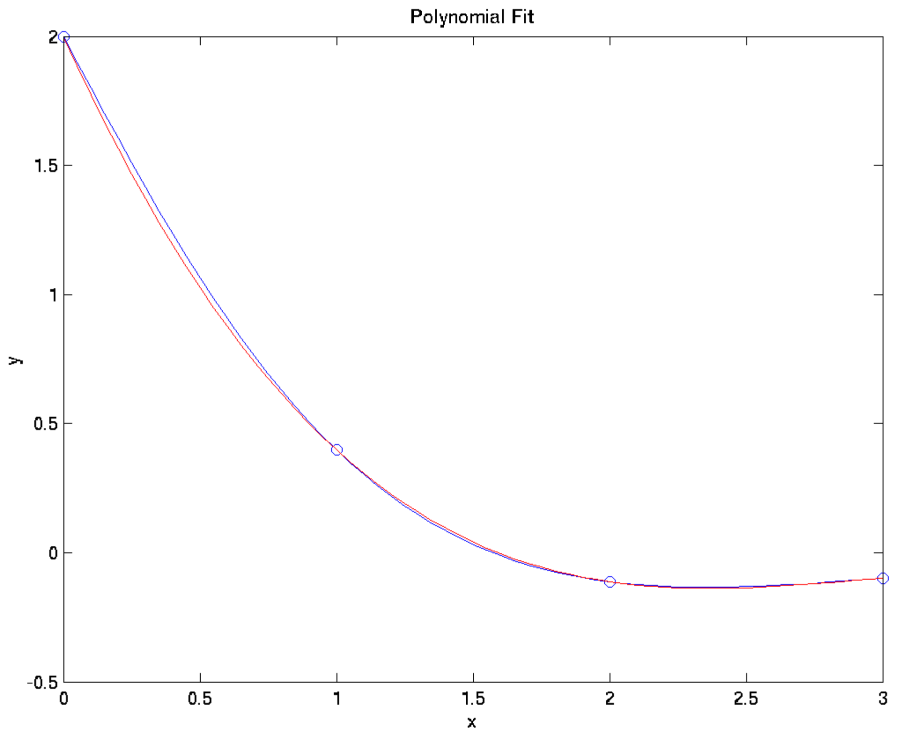

Figure 1: Four points (o) with cubic polynomial fit (red) and exact (blue)

2.2 Least Squares Fit

Instead of high order polynomials, it is often sufficient to find a low order polynomial that goes through the data in a "best fit" manner. For N points, we will have a fitting polynomial of order m, which is often considerably smaller than N. The least squares fitting polynomial will be similar to the exact fit form: $\mathbf{y_p}(\mathbf{x}) = \mathbf{P}\,\mathbf{a}$, where $\mathbf{P}$ has the same form as before (Eqn. 2), except that it is an N x m matrix, as we have more equations than unknowns, and this equation is not directly invertable for the unknown coefficient $\mathbf{a}$.

If we look at the error between the best fit interpolating polynomial and the data, defined as $\epsilon_i = |y_P(x_i) - y_i|$, then one method of finding a polynomial is to chose a polynomial of order 2 or 3, then minimize the sum of all the errors, $\sum_{i=1}^{N} \epsilon_i^2 = (\mathbf{Pa} - \mathbf{y})^2$, with respect to the unknown coefficients of the interpolant. This gives

$$\begin{aligned} 2\mathbf{P}^T\mathbf{Pa} - 2\mathbf{P}^T\mathbf{y} &= 0 && (5)\\ \text{Solving, we have } \mathbf{a} &= \left(\mathbf{P}^T\mathbf{P}\right)^{-1}\mathbf{P}^T\,\mathbf{y}^T && (6)\end{aligned}$$

The solution can also be written as $\mathbf{a} = \mathbf{P}^{+}\mathbf{y}^T$, where $\mathbf{P}^{+}$ is known as a *pseudoinverse.* When, as in exact polynomial fitting, $\mathbf{P}$ is a square matrix, then the pseudoinverse is the same as the inverse, and the above solution reduces to Eqn. 4.

2.2.1 Problem

Show that the best fit straight line through the data (x_i, y_i) for $i = 1, \ldots, N$, satisfies the following equations by deriving them from Eqn. 6.

$$y_p(x) = mx + b \tag{7}$$

where

$$m = \frac{\left(N \sum_i^N x_i y_i - \sum_i x_i \sum_i y_i\right)}{N \sum_i x_i^2 - \left(\sum_i x_i\right)^2} \tag{8}$$

$$b = \frac{\sum_i^N y_i - m \sum_i x_i}{N} \tag{9}$$

2.3 Piecewise Polynomials

So far, we have interpolated data with N^{th} order polynomials through $N+1$ points or m^{th} order $(m < N)$ polynomials that best fit the data. Another approach is to use piecewise polynomials, which are fit between each pair of points. For example, we can connect each of the data points with straight lines. This gives us interpolants that ensure that the given data at a point is correctly approached from either direction. For more accuracy, higher order functions are used, such as quadratic or cubic polynomials. This family of piecewise polynomials is referred to as *splines.* The cubic spline is very popular and guarantees that the slope and the second derivative is continuous on both sides of a data point.

The next types of interpolants are more sophisticated forms for local fits to the data.

2.4 Radial Basis Functions

In a number of fields, it is important to interpolate between irregularly spaced two or three-dimensional data. In coastal/ocean engineering, imagine an offshore area with randomly distributed sounding data. How do you obtain a Cartesian grid of 2D depth values for use in a wave model?

One such method is through the use of *radial basis functions*, Hardy (1971, 1990); Kansa (1990). These functions, which are functions only of distance from a point, have a variety of shapes; for now, we'll use the function $g(x)$. Wendland (2005) points out some of the advantages of RBFs: they are useful for problems for any number of spatial dimensions; they are applied to data of arbitrarily scattered data, the interpolants are of arbitrary smoothness, and they have a simple structure.

For a given set of N points in two or three dimensions, $\mathbf{x}_i, i = 1, N$, (we could call the points nodes instead), we will have a set of RBF basis functions: $g_j(\mathbf{x}) \equiv g(|\mathbf{x} - \mathbf{x}_j|), j = 1, N$, where $|\mathbf{x} - \mathbf{x}_j|$ is the distance from the position $\mathbf{x}$ to the node $\mathbf{x}_j$. Thus there is a radial basis function for each $\mathbf{x}_i$. For each of these given points, we have the values $u_i, i = 1, N$, where $u(\mathbf{x})$ is the dependent variable that is to be interpolated. The RBF interpolant is then defined as

$$u_p(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \; g_j(\mathbf{x}) \tag{10}$$

To find the unknown coefficients $\alpha_j, j = 1, N$, we use collocation to force the interpolant to pass through the given data:

$$\sum_{j=1}^{N} \alpha_j \; g_j(\mathbf{x}_i) = u_i, \quad i = 1, N \tag{11}$$

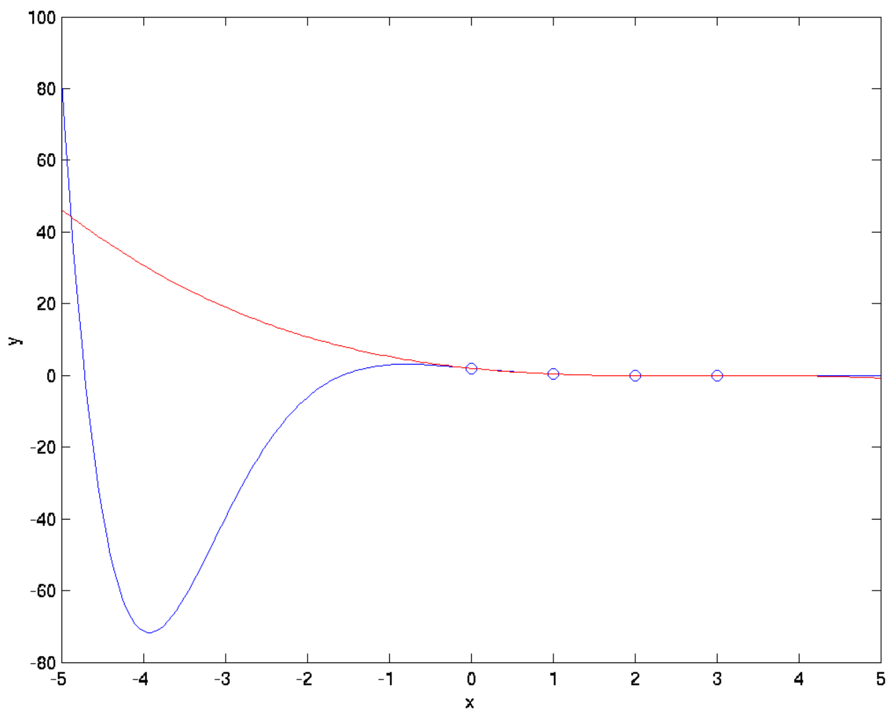

Figure 2: Four points (o) with cubic polynomial fit (red) and exact (blue) in extended range.

These N equations constitute a matrix equation to be solved for the N values of the α_j.

There is an infinite variety of functions to use for the RBF bases. Examples are:

$$\begin{array}{ll}
\text{Multiquadrics (MQ)} & \sqrt{c_j^2 + r^2} \\[2ex]
\text{Reciprocal MQ} & \frac{1}{\sqrt{c_j^2+r^2}} \\[2ex]
\text{Gaussians} & e^{-c^2 r^2} \\[2ex]
\text{Thin plate splines} & r^2 \log(r)
\end{array} \tag{12}$$

where $r = |\mathbf{x} - \mathbf{x}_j|$. The parameters c_j can either be a constant or vary with nodal position. They are added for scaling and to ensure that the RBF is bounded at the points. Choosing the best value requires some experimentation. Examples of these RBFs are shown in Figure 3.

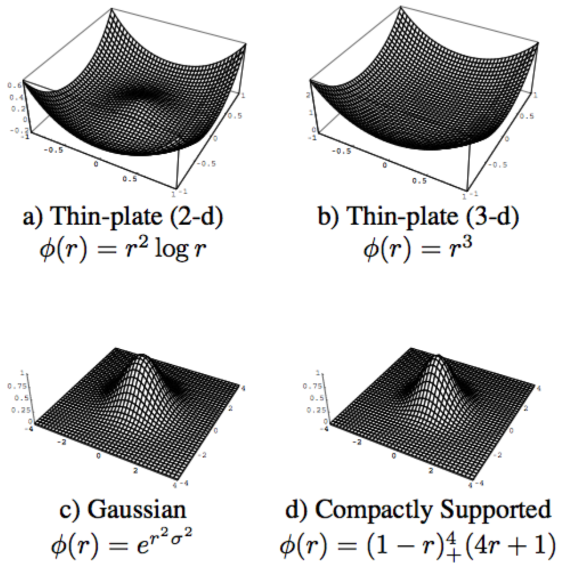

Figure 3: Four RBFs, from Morse *et al.* (2001)

Buhman (2000) points out that the success of the RBF method can be attributed to the fact that many of the RBF functions have the form of the Greens function for many differential equations.

Example 2 The input data are irregularly spaced on the x axis and are the following 11 (x,y) pairs: (0.2, 3.00), (0.38, 2.10), (1.07, -1.86), (1.29, -2.71), (1.84, -2.29), (2.31, 0.39), (3.12, 2.91), (3.46, 1.73), (4.12, -2.11), (4.32, -2.79), (4.84, -2.25). Both the MQ interpolations and the reciprocal interpolants are determined for these data. The results are shown for the RMQ in Figure 4. The data actually correspond to $y(x) = 3\ \sin(2\pi x/3)$. In the figure, the input data are shown with large Xs, the interpolation (at a finer resolution) with + signs and the exact function is plotted with a line. Clearly the fit is quite good.

The influence of the value of c used in the multiquadric method is shown in the two logarithmic root-mean-square error plots shown in Figures 5 and 6. Of the two, the reciprocal multiquadric is not as accurate as the regular multiquadric.

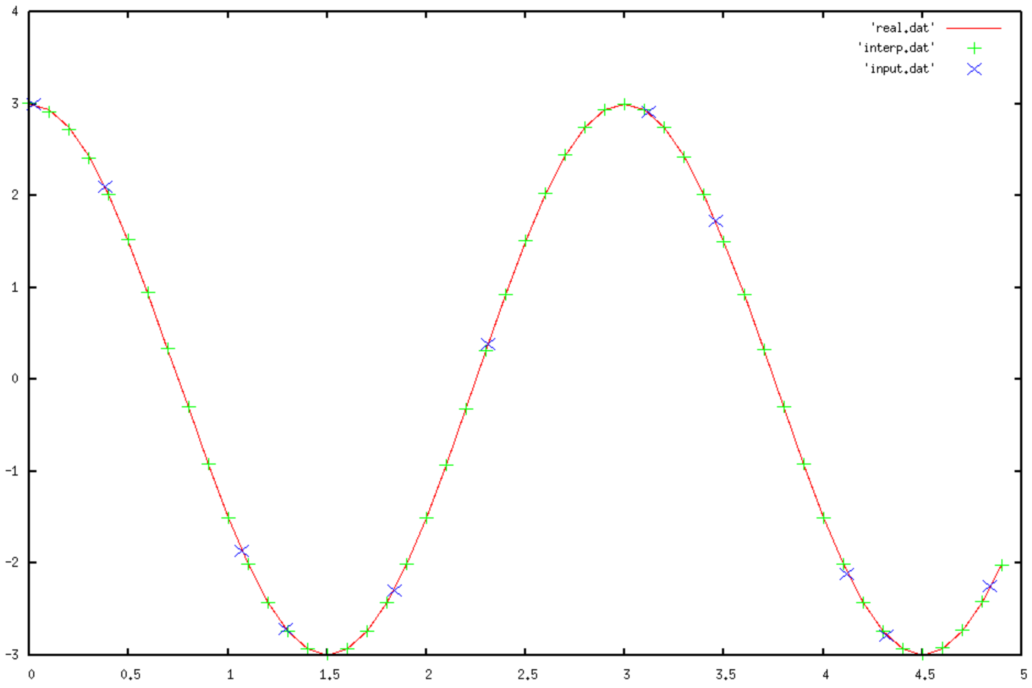

Figure 4: A reciprocal multiquadric fit to the given data.

Problem Use an RBF to find the interpolating function for the four points given in Example 1. Use the reciprocal multiquadric form. A "good" answer with $c = 0.8$ and the reciprocal multiquadric method gives the weights of the RBFs located at the four given points in Example 1 as 2.295, -0.87, -0.299, and -0.160.

2.4.1 Partial Differential Equations

Now let's examine a partial differential equation. For simplicity, the equation will be the Laplace equation in two dimensions with imposed Dirichlet (essential) boundary conditions, Kansa (1999). This is stated as

$$\nabla^2\phi(x,y) = 0 \quad (13)$$

$$\phi(x,y)\big|_{\text{on the boundary}} = f(x,y) \quad (14)$$

The solution proceeds by choosing points within the fluid domain and then points on the boundary. The RBF solution will be sought in the form $\phi_h(\mathbf{x}) = \sum_{j=1,N} \alpha_j g_j(\mathbf{x})$, where N is the total number of points.

Substituting into the Laplace equation and the boundary condition using collocation again gives

$$\sum_{j=1}^{N} \alpha_j \nabla^2 g_j(x_i) = 0 \text{ for all the interior points} \quad (15)$$

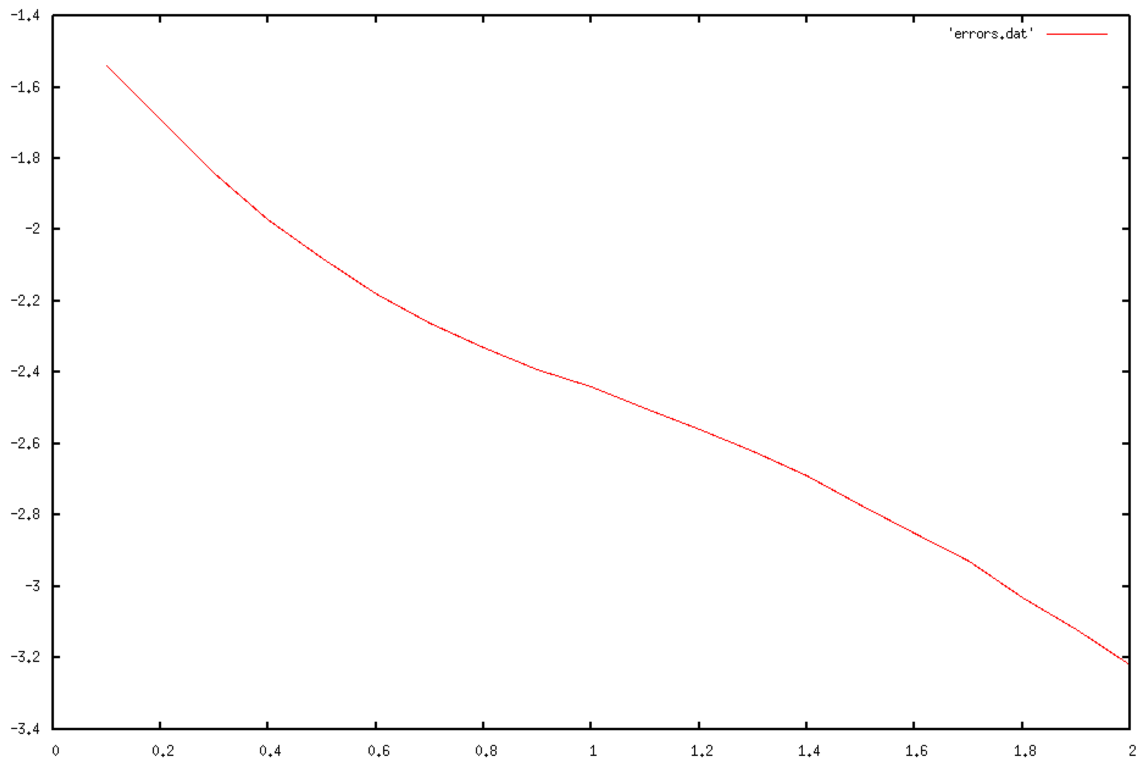

Figure 5: Log_{10} (root-mean-square error) as a function of c for the multiquadric method.

$$\sum_{j=1}^{N} \alpha_j g_j(x_i) \quad = \quad f_i \text{ for the boundary points} \tag{16}$$

This N x N matrix equation is solved for the N values of α.

The resulting solution is a multiquadric function for ϕ. There is just one set of coefficients for the domain.

The drawback of this method is that the matrix that must be solved is a full matrix with contributions from all the RBFs; that is, it is N x N, where N is the number of fixed points. The fact that the points $\mathbf{x}_i$ are fixed would prove to be a problem for problems where the domain changes with time, such as free surface problems where the water surface detaches–say in wave breaking, and the fluid leaves the computational domain (thus requiring some clever programming).

One way to avoid computing the full matrix with contributions from each of the RBFs is to choose radial basis functions with compact support; that is, functions that do not exist over the entire domain, but only within a small distance of given point. These functions would only have a local influence. Wendland (2005) provides a number of these basis functions. Two examples are

$$\text{1D} : \begin{cases} (1 - r/h)^3(3r/h + 1) & \text{for } |r| < h \\ 0, \text{ otherwise} \end{cases} \tag{17}$$

$$\text{3D} : \begin{cases} (1 - r/h)^4(4r/h + 1) & \text{for } |r| < h \\ 0, \text{ otherwise} \end{cases} \tag{18}$$

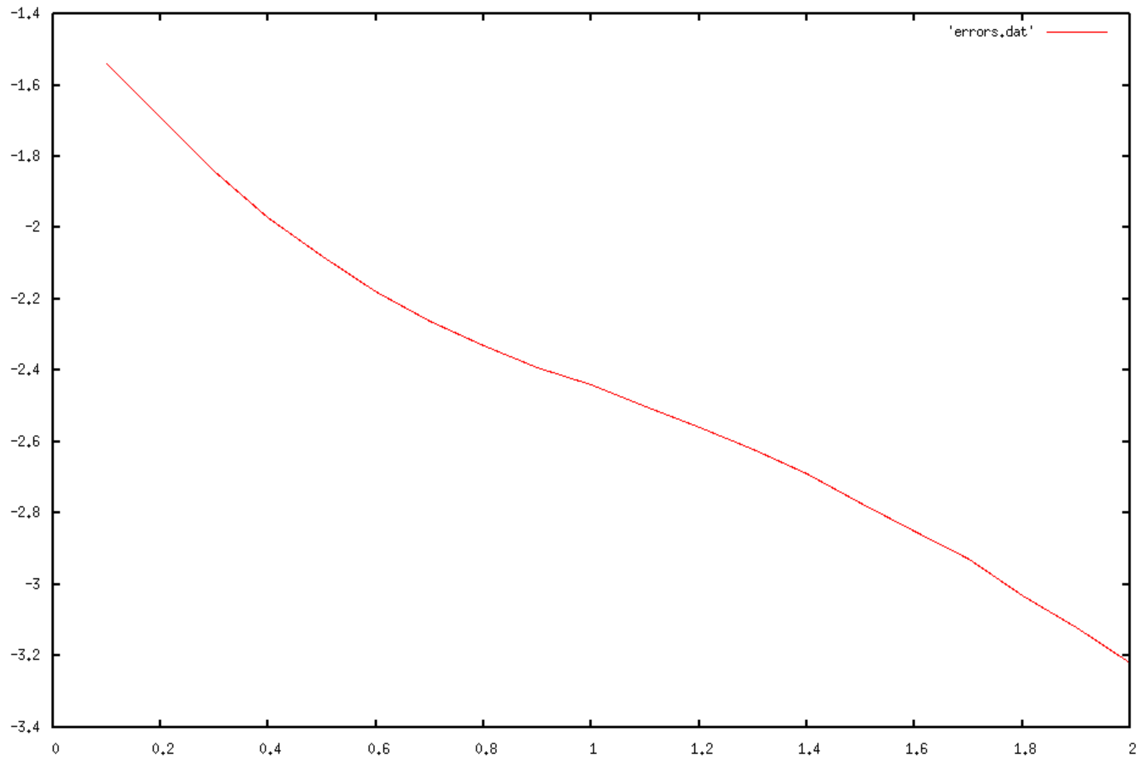

Figure 6: Log_{10}(root-mean-square error) as a function of c for the reciprocal multiquadric method.

The parameter h determines the size of the RBF. The 1D Wendland RBF is shown in Figure 7. The computational procedure, which saves time, is to determine, for each location, $\mathbf{x}_i$, which of the RBFs contribute, so those that do not contribute are not calculated. This saves time for large problems. Further, the resulting matrix is sparse.

The Wendland 1D radial basis function is referred to in the meshless methods literature as a quartic spline weight function, while his 3D RBF is the quintic spline weight function .

2.4.2 Consistency

Consistency is the ability of an interpolating polynomial to reproduce a polynomial of a given order. The simplest consistency is constant consistency. For example, if at all the data points $\mathbf{x}$ the value is unity, then we have

$$\sum_{j=1}^{N} \alpha_j \; g_j(\mathbf{x}_i) = 1, \;\; i = 1, N \tag{19}$$

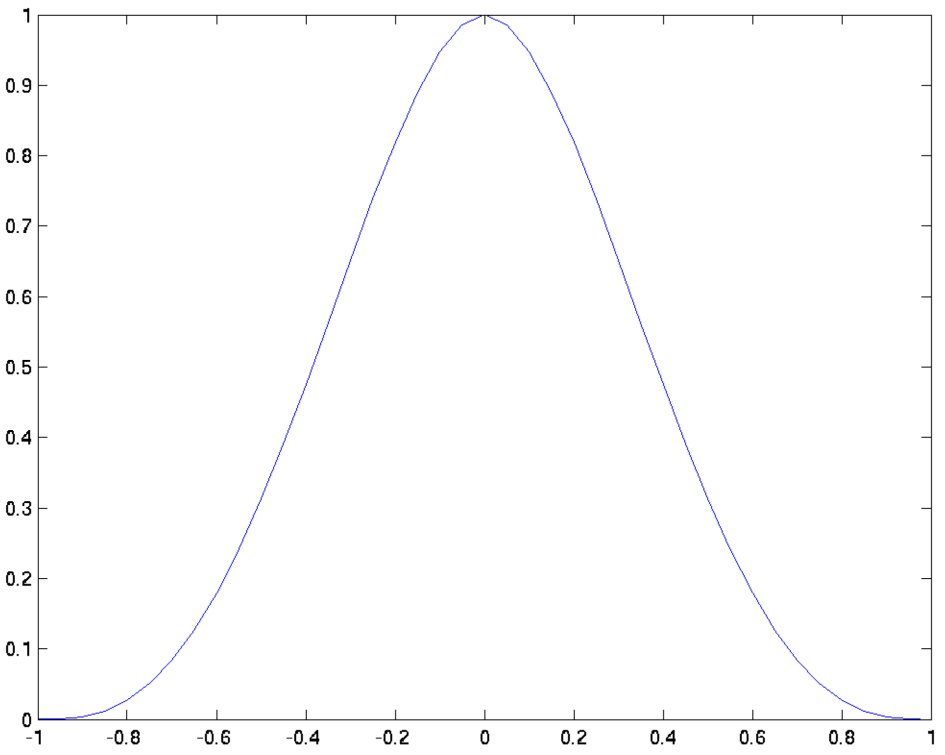

Figure 7: Wendland's (2005) 1D Radial Basis Function, with h=1.

If as in Gaussian RBFs and Wendland's RBF, the value of the RBF at its center of support is unity, then we have the following constraint.

$$\sum_{j=1}^{N} \alpha_j = 1$$

For other RBFs, they would have to be normalized to give unity at the $r = 0$.

2.5 Moving Least-Squares Interpolant

Moving Least-squares interpolants (MLS) are another interpolating scheme for irregularly spaced data, $u(\mathbf{x}_i)$, for $i = 1, \ldots, N$, in several dimensions, but it has the distinction of providing a fitting function $g(\mathbf{x})$ that does not explicitly go through the given data points as the RBF collocated functions did, Lancaster and Salkauskas (1981); furthermore the interpolant at a given location is best fit to only nearby points.

The interpolating function at any value of $\mathbf{x}$ (following Belytschko *et al.* (1994); Dilts (1999)) is

$$u_p(\mathbf{x}) = \sum_j^N a_j(\mathbf{x}) p_j(\mathbf{x}) \equiv \mathbf{p}^T(\mathbf{x})\, \mathbf{a}(\mathbf{x}) \tag{20}$$

where $p^T(\mathbf{x})$ are monomials in the coordinate $\mathbf{x}$. For example, in one dimension, $p^T(\mathbf{x}) = [1, x]$ and $N = 2$ for a linear approach, or $p^T(\mathbf{x}) = [1, x, x^2]$ and $N = 3$ for a quadratic approach. In two dimensions, the corresponding linear and quadratic bases are

$$\begin{aligned} p^T(\mathbf{x}) &= [1, x, y], \quad \text{N= 3} & (21)\\ p^T(\mathbf{x}) &= [1, x, y, x^2, xy, y^2], \;\; \text{N} = 6 & (22) \end{aligned}$$

The coefficients of the interpolant, $\mathbf{a}(\mathbf{x})$ vary with $\mathbf{x}$, hence the name "moving," as the interpolant is constantly changing.

To find these coefficients at position $\mathbf{x}$, the sum of the weighted squared errors between the interpolating function for each position $\mathbf{x}_i$ and data u_i is computed:

$$E(\mathbf{x}) = \sum_{i=1}^N W(\mathbf{x} - \mathbf{x}_i) \left(\mathbf{p}^T(\mathbf{x}_i)\, \mathbf{a}(\mathbf{x}) - u_i\right)^2 \tag{23}$$

where $W(\mathbf{x} - \mathbf{x}_i)$ is a weighting function for node i that decays with the distance from the node to position $\mathbf{x}$. Because of the weighting functions, contributions to the sum from distant nodes is either small or zero.

Minimizing the mean-squared error with respect to the coefficients, we obtain

$$\frac{\partial E}{\partial \mathbf{a}} = \mathbf{A}(\mathbf{x})\mathbf{a}(\mathbf{x}) - \mathbf{B}(\mathbf{x})\mathbf{u} = 0 \tag{24}$$

where

$$\begin{aligned} \mathbf{u}^T &= (u_1, u_2, ...u_n) & (25)\\ \mathbf{P} &= \begin{bmatrix} p_1(\mathbf{x}_1) & p_2(\mathbf{x}_1) & \dots & p_m(\mathbf{x}_1) \\ p_1(\mathbf{x}_2) & p_2(\mathbf{x}_2) & \dots & p_m(\mathbf{x}_2) \\ \dots & \dots & \dots & \dots \\ p_1(\mathbf{x}_n) & p_2(\mathbf{x}_n) & \dots & p_m(\mathbf{x}_n) \end{bmatrix} & (26)\\ \mathbf{W}(\mathbf{x}) &= \begin{bmatrix} W(\mathbf{x} - \mathbf{x}_1) & 0 & \dots & 0 \\ 0 & W(\mathbf{x} - \mathbf{x}_2) & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & W(\mathbf{x} - \mathbf{x}_n) \end{bmatrix} & (27)\\ \mathbf{A} &= \mathbf{P}^T \mathbf{W}(\mathbf{x}) \mathbf{P} & (28)\\ \mathbf{B} &= \mathbf{P}^T \mathbf{W}(\mathbf{x}) & (29) \end{aligned}$$

Solving, $\mathbf{a}(\mathbf{x}) = \mathbf{A}^{-1}(\mathbf{x})\ \mathbf{B}(\mathbf{x})\, \mathbf{u}$. The local approximation at $\mathbf{x}$ is then given by Eqn. 20. For example, with N data points, $m = 3$, and p^T given as in Eq. 21, we have

$$\begin{aligned} \mathbf{A} &= \begin{pmatrix} \sum_i W(\mathbf{x} - \mathbf{x}_i) & \sum_i W(\mathbf{x} - \mathbf{x}_i)x_i & \sum_i W(\mathbf{x} - \mathbf{x}_i)y_i \\ \sum_i W(\mathbf{x} - \mathbf{x}_i)x_i & \sum_i W(\mathbf{x} - \mathbf{x}_i)x_i^2 & \sum_i W(\mathbf{x} - \mathbf{x}_i)x_i y_i \\ \sum_i W(\mathbf{x} - \mathbf{x}_i)y_i & \sum_i W(\mathbf{x} - \mathbf{x}_i)x_i y_i & \sum_i W(\mathbf{x} - \mathbf{x}_i)y_i^2 \end{pmatrix} & (30)\\ \mathbf{B} &= \begin{pmatrix} W(\mathbf{x} - \mathbf{x}_1) & W(\mathbf{x} - \mathbf{x}_2) & W(\mathbf{x} - \mathbf{x}_3) & \dots & W(\mathbf{x} - \mathbf{x}_N) \\ W(\mathbf{x} - \mathbf{x}_1)x_1 & W(\mathbf{x} - \mathbf{x}_2)x_2 & W(\mathbf{x} - \mathbf{x}_3)x_3 & \dots & W(\mathbf{x} - \mathbf{x}_N)x_N \\ W(\mathbf{x} - \mathbf{x}_1)y_1 & W(\mathbf{x} - \mathbf{x}_i)y_2 & W(\mathbf{x} - \mathbf{x}_i)y_3 & & W(\mathbf{x} - \mathbf{x}_N)y_N \end{pmatrix} & (31) \end{aligned}$$

For the same irregular 1D data from Example 2, we will use the MLS method to find interpolated values at different points. The weighting function chosen will be a cubic spline of the following form:

$$W(q) = \frac{2}{3h} \begin{cases} 1 - \frac{3}{2}q^2 + \frac{3}{4}q^3, & \text{for } q \leq 1 \\ \frac{1}{4}(2-q)^3, & \text{for } 1 \leq q \leq 2 \\ 0, & \text{for } q > 2 \end{cases} \tag{32}$$

The argument q is the normalized distance $(x - x_i)/h$, where h determines the size of the support for $W(q)$ as the value of $W(q)$ goes to zero for $q > 2h$. This means that given data within a range of $\pm 2h$ of a given point will be used. So, again, the MLS is a local fit to the data. Figure 8 shows the shape of the cubic spline. This weighting function is used also in many Smoothed Particle Hydrodynamics models.

The choice of the weight function, it turns out, is not that important, except that it must be positive everywhere and decrease monotonically in magnitude with distance. It also should be differentiable for as many times as needed.

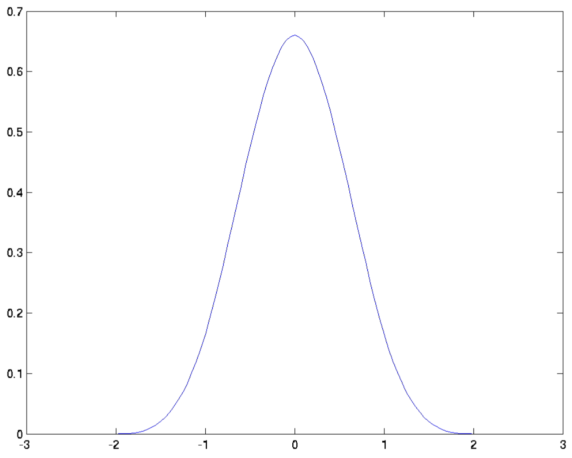

Figure 8: Cubic spline Weighting Function (h=1 for this case), as defined in Eqn. 32

The size h of the support for the weighting function is important. For rapidly varying data, large h values lead to excessive averaging. For a linear interpolant, only a region of data where the variation is linear should be included.

The results of a moving least squares interpolation are no longer a single polynomial, but a different polynomial fit for each point $\mathbf{x}$. However, it can be shown that data provided by a polynomial of the same degree as the monomial basis function is reproduced exactly by the method. Also, if the weight function is differentiable r times, then the resulting MLS polynomials are also differentiable r times. Finally, if the weighting function is taken as unity everywhere, then the results of the MLS is the least squares fit of the data.

Example 3

A moving least squares interpolant, using the cubic spline weighting function ($h = 0.51$), is shown in Figure 9. In the figure, the solid line is the function from which the data, shown with circles, were obtained: $y(x) = 3\cos(2\pi\ x/3)$. The asterisks show the interpolated values. Note mean squared error is 0.0051. The h value in this problem had to be kept small due to the rapid variation of the data with x, as otherwise there is too much smoothing. Figure 10 show the effect of the weighting function on the ability of the MLS to fit the sinusoidal data; the best fit is for h on the order of 0.5, which is about 1/6 of the wave length of the data and also, more importantly for this example, about the average spacing between given data points.

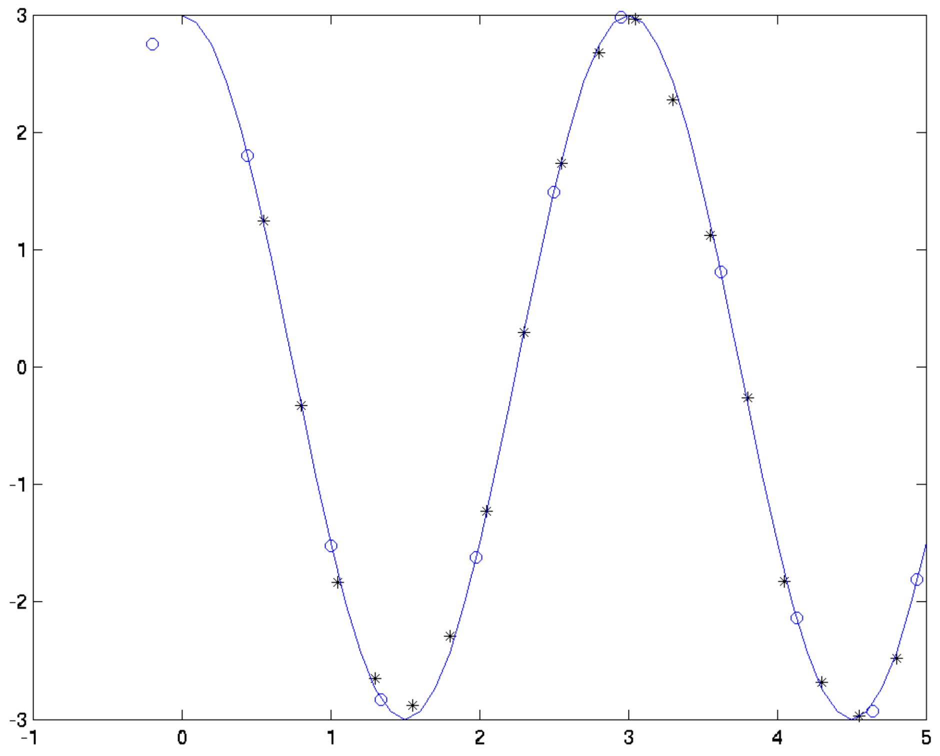

Figure 9: Moving Least Square fit to randomly spaced data, as in Example 2. Solid line is the function that generated the data (shown by circles). MLS interpolant given by Eqn. 20 and the * symbol.

3 Smoothed Particle Hydrodynamics

In the previous section, we have seen how point or nodal methods can lead to effective interpolation scheme and to methodologies for solving differential equations. These methods have the advantage that they don't require meshes as in finite elements, where meshing can be rather complicated and expensive. Drawbacks of

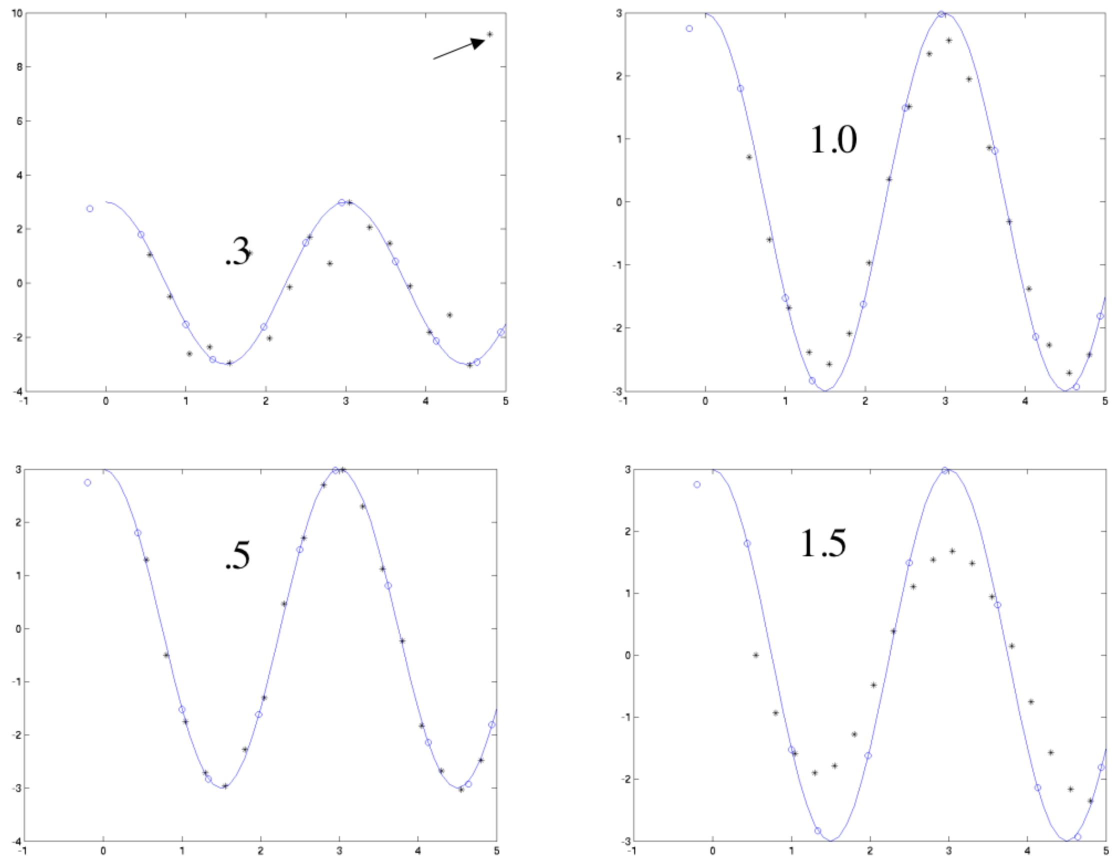

Figure 10: Moving Least Square fit to randomly spaced data, as in the previous figure, as a function of the value of h. Solid line is the function that generated the data (shown by circles). Large values of h result in too much smoothing of variable data.

the previous methods, such as radial basis functions, are that they involve fixed particles and they require matrices that are N x N (N = number of points) to be solved. For thousand of points, that's a big matrix. Further for high resolution, many hundreds of thousands of points are needed. While radial basis functions with compact support lead to sparse matrices, they still are large.

Smoothed particle hydrodynamics was begun by Lucy (1977) and Gingold and Monaghan (1977) for use in astrophysics. Review articles by Benz (1990) and Monaghan (1992) provide a good historical background.

For thirty years SPH has been a major numerical tool for the evolution of galaxies, galactic collisions, and bolid impact. Since then, it has been modified for use in solid mechanics (such as impact problems), and fluid mechanics (such as casting in molds). In 1994, Monaghan (1994) showed that SPH could be used for free surface flows and it had the advantage that no special treatment was needed at the free surface; that is, no imposed boundary conditions.

A conceptual view of SPH is that the fluid flow is decomposed into parcels of fluid. Each of these parcels (particles) has an associated mass, density, velocity, pressure, etc, and that these values are constant for that parcel. Using the information from the parcels we can interpolate values of these variables to any location within the fluid. Using the interpolating function, we can obtain derivatives of variables at any location as well. Due to pressure differences between the parcels (a pressure gradient), the particles move with time, carrying along their variable values, which can change with time. The SPH model then follows the trajectory of each of the (thousands of) particles. The sum of all the particle motions is a flow. (Surprisingly an SPH model with one particle will show that the particle acts as an elastic solid–bouncing from walls, which fluids do not do).

The SPH method is based on the following integral

$$u(\mathbf{s}, t) = \int_{\upsilon} W(\mathbf{s} - \mathbf{x}, h)\ u(\mathbf{x}, t)\ d\upsilon \tag{33}$$

where the integral is over the domain υ and $d\upsilon$ is $dx\,dy\,dz$ in three dimensions and $dx\,dy$ in two, or dx in one dimension. The size of the support for the weighting function, $W(\mathbf{x}, h)$, is determined by the parameter h, which is a circle of radius $2h$ in 2D and a sphere of the same radius in 3D. In the simplest application of SPH, h is a constant; however, it is relatively straightforward to have h be a function of $\mathbf{x}$ and of time.

Analytically, the definition of a delta function $\delta(\mathbf{x})$, is such that

$$\int_{\upsilon} f(\mathbf{x})\, \delta(\mathbf{x} - c)\, d\mathbf{x} \quad = \quad f(c) \tag{34}$$

$$\int_{\upsilon} \delta(\mathbf{x} - c)\, d\mathbf{x} \quad = \quad 1 \text{ for any } c \tag{35}$$

An example of a delta function in one dimension is given by Li and Liu (2004):

$$\delta(x) = \lim_{\epsilon \to 0} \begin{cases} 0, & x < -\epsilon/2 \\ 1/\epsilon, & -\epsilon/2 < x < \epsilon/2 \\ 0, & x > \epsilon/2 \end{cases} \tag{36}$$

Therefore if the weighting function, usually called a *kernel*, behaves like a δ function, then Eq. 33 is valid. A kernel that makes the equation true is often called a *reproducing kernel.*

For *consistency*, or what is known in the finite element literature as *completeness*, this integral should be correct for a constant. Substituting $u(\mathbf{x}, t) = 1$ into Eq 33 yields

$$\int_{\upsilon} W(\mathbf{s} - \mathbf{x}, h)\ d\upsilon = 1 \tag{37}$$

which means that the equation reproduces constants correctly. Monaghan (1992) has a second requirement for the kernel,

$$\lim_{h \to 0} W(\mathbf{x}, h) = \delta(\mathbf{x}) \tag{38}$$

To use these equations, they have to be approximated numerically by a summation of contributions from the particles in the domain:

$$u(\mathbf{s}, t) \approx \sum_j W(\mathbf{s} - \mathbf{x}_j, h)\ u_j\ \Delta v_j \tag{39}$$

$$\sum_j W(\mathbf{s} - \mathbf{x}_j, h)\ \Delta v_j = 1 \tag{40}$$

where Δv_j is the incremental volume (or area in 2D) associated with particle j. In SPH, the mass of particle j is fixed as m_j and the density in the vicinity of the particle ρ_j is allowed to vary (remember, it is a compressible fluid). In this expression u_j, m_j and ρ_j are not functions of space, and m_j is not a function of time.

Therefore we can replace the volume Δv_j with m_j/ρ_j. We then rewrite the last equation as

$$u(\mathbf{s}, t) \approx \sum_j \frac{m_j}{\rho_j}\ W(\mathbf{s} - \mathbf{x}_j, h)\ u_j(t) \tag{41}$$

This equation is the interpolation equation associated with SPH. We can interpolate the value of u at $\mathbf{x} = \mathbf{s}$ with this equation. In that sense, it is, at this stage, similar to all the other interpolation methods discussed earlier. In fact, we could rewrite the expression more succinctly as

$$u(\mathbf{s}) = \sum u_j g_j(\mathbf{x})$$

where $g_j = m_j W(\mathbf{s} - \mathbf{x}, h)/\rho_j$.

3.1 Kernels

A variety of kernels have been used with SPH. Most satisfy the above conditions for kernels, plus they are always positive. Gingold and Monaghan (1977) used a Gaussian kernel, which is

$$W(\mathbf{x}, h) = \alpha\ e^{-q^2} \tag{42}$$

where $\alpha = 1/(h\sqrt{\pi}), 1/(h^2\pi),$ and $1/(h^3\pi^{3/2})$ in one, two, and three dimensions, and again $q = |\mathbf{x} - \mathbf{x}_i|/h$. This kernel satisfies both of the requirements for the kernel (Eqns. 34 and 35); however, it does not have compact support. Another kernel is used in the method Moving Particle Semi-implicit method (MPS; to be discussed later):

$$W(\mathbf{x} - \mathbf{x}_i, h) = \begin{cases} \frac{1}{q} - 1, & q < 1 \\ 0, & q > 1 \end{cases} \tag{43}$$

This kernel has a discontinuity in slope at $q = 1$.

The quadratic kernel, Johnson *et al.* (1996), is

$$W(\mathbf{x} - \mathbf{x}_i, h) = \begin{cases} \beta\left(\frac{1}{4}q^2 - q + 1\right), & q < 2 \\ 0, & q > 2 \end{cases} \tag{44}$$

where $\beta = 3/(4h), 3/(2\pi h^2),$ and $15/(16\pi h^3)$ in 1, 2 and 3D. One of the advantages of the quadratic kernel is that there is no maximum in the first derivative of the kernel away from the origin.

The cubic spline kernel was defined in Eq. 32. The quartic spline, which is the same as Wendland's 1D RBF, is given by

$$W(\mathbf{x} - \mathbf{x}_i, h) = \begin{cases} \alpha\left(1 - 6q^2 + 8q^3 - 3q^4\right), & q < 1 \\ 0, & q > 1 \end{cases} \tag{45}$$

while the quintic spline weighting function is

$$W(\mathbf{x} - \mathbf{x}_i, h) = \begin{cases} \alpha\left(1 - 10q^2 + 20q^3 - 15q^4 + 4q^5\right), & q < 1 \\ 0, & q > 1 \end{cases} \tag{46}$$

where the αs are chosen to ensure that the integral of the kernel over the domain is unity.

Higher order completeness of the kernels (that is, their ability to reproduce polynomials) can be determined by examining Eq. 33 in more detail (Liu and Liu (2003), also see their Table 3.1). Restricting ourselves to 1D and expanding $u(x,t)$ in a Taylor series in x, we have

$$\begin{aligned} u(s,t) &= \int_{\mathcal{V}} W(s-x,h)\; u(x,t)\; dx \qquad (47) \\ &= \int_{\mathcal{V}} W(s-x,h)\; \left(u(s,t) + u'(s)(x-s) + \frac{1}{2}u''(s)(x-s)^2 + \ldots\right) dx \qquad (48) \\ &= u(s,t)\int_{\mathcal{V}} W(s-x,h)\; dx + u'(s)\int_{\mathcal{V}} W(s-x,h)(x-s)\; dx + \qquad (49) \\ &\quad \frac{1}{2}u''(s)\int_{\mathcal{V}} W(s-x,h)(x-s)^2\; dx \ldots \end{aligned}$$

To ensure the equality is true for each order kept in the Taylor series expansion, we have the following requirements:

$$\begin{aligned} \int_{\mathcal{V}} W(s-x,h)\; dx &= 1 \qquad (50) \\ \int_{\mathcal{V}} W(s-x,h)\; (x-s)\; dx &= 0 \qquad (51) \\ \int_{\mathcal{V}} W(s-x,h)\; (x-s)^2\; dx &= 0 \qquad (52) \end{aligned}$$

The first condition we have seen before, Eq. 37; the second is satisfied by any kernel that is symmetric. Since all the kernels above are symmetric, we argue that SPH has first order completeness. However, this is in a formal sense, we have to verify that with the numerical approximations to the integrals.

$$\begin{aligned} \sum_j W(\mathbf{s} - \mathbf{x}_j, h)\; \Delta v_j &= 1 \text{ as before,} \qquad (53) \\ \sum_j W(\mathbf{s} - \mathbf{x}_j, h)\; (\mathbf{x} - s)\; \Delta v_j &= 0 \qquad (54) \\ \sum_j W(\mathbf{s} - \mathbf{x}_j, h)\; (\mathbf{x} - s)^2\; \Delta v_j &= 0 \qquad (55) \end{aligned}$$

are the conditions for constant, linear, and second order consistency (completeness). Note that the summations are taken over the particles contained within the support of the kernel (that is, within $2h$ of the point of interest, s). Also, the first of these equations is sometimes referred to as a *partition of unity.*

One method of ensuring the constant consistency can be obtained for the case that all particles have the same associated volume element, $\Delta\upsilon$. From the first of the above equations, we can solve for the volume element:

$$\Delta\upsilon = 1/\sum_j W(\mathbf{s}-\mathbf{x}_j,h) \tag{56}$$

then substituting into Eq. 39 gives

$$u(\mathbf{s},t) = \sum_j \frac{W(\mathbf{s}-\mathbf{x}_j,h)}{\sum_j W(\mathbf{s}-\mathbf{x}_j,h)}\, u_j(t) \tag{57}$$

This normalization of the kernel by the summation term is known as the Shepard interpolant.

Example 3 MLS Approximation Derivation

The Shepard interpolant can be derived directly from the moving least-squares interpolant using only the constant in the monomial basis $\mathbf{p}$. Referring to Section 2.5, the interpolant at a position $\mathbf{x}$ is $u_p(\mathbf{x}) = \sum_j^N a_j(\mathbf{x})p_j(\mathbf{x}) \equiv \mathbf{p}^T(\mathbf{x})\,\mathbf{a}(\mathbf{x})$ from Eqn. 20. The terms were defined as

$$\begin{aligned}\mathbf{A} &= \mathbf{P}^T\mathbf{W}(\mathbf{x})\mathbf{P}\\ \mathbf{B} &= \mathbf{P}^T\mathbf{W}(\mathbf{x})\end{aligned}$$

but here $\mathbf{p}^T = [1\ 1\ 1\ldots]$. Substituting, we find that

$$\mathbf{a} = \frac{\sum_j W_j u_j}{\sum_j W_j} \tag{58}$$

and

$$u_i = \frac{\sum_j W_j u_j}{\sum_j W_j} \tag{59}$$

This is the same as in Eq. 57.

3.1.1 Gradients

One of the powerful advantages of the SPH kernel approach is that the derivative of a function is calculated analytically, as compared to a method like finite differences, where the derivatives are calculated from neighboring points using the spacing between them. For the irregularly spaced SPH particles, this would be extremely complicated.

To take the derivative of $u(\mathbf{x},t)$ with respect to x, we have

$$\nabla u(\mathbf{s},t) = \int_\upsilon u(\mathbf{x},t)\nabla W(\mathbf{s},t)\, d\upsilon \tag{60}$$

Liu and Liu (2003) show that consistency requires that the kernel satisfy the following conditions (in one D):

$$\int_{\upsilon} W'(s-x,h)\,dx = 0 \tag{61}$$

$$\int_{\upsilon} W'(s-x,h)\,(x-s)\,dx = 1 \tag{62}$$

$$\int_{\upsilon} W'(s-x,h)\,(x-s)^2\,dx = 0 \tag{63}$$

Most SPH formulations do not meet these conditions.

In particle form, the gradient is

$$(\nabla \mathbf{u})_j = \sum_i \frac{m_i}{\rho_i}\, u_i\, \nabla W(x_j - x_i) \tag{64}$$

3.2 Governing Equations

In smoothed particle hydrodynamics, the fluid has been traditionally considered compressible. The reason is that it is easier to calculate the pressure from an equation of state rather than having to solve for the pressure. For large changes in pressure, water is compressible, but for coastal applications we can consider it to be incompressible. We will get around the issue of compressibility by assuming that the water is only slightly compressible.

3.2.1 Conservation of Mass

The conservation principle for a compressible fluid is, in Eulerian form,

$$\frac{\partial \rho}{\partial t} + \nabla \rho \mathbf{u} = 0 \tag{65}$$

In Lagrangian form, following a parcel of fluid, we have that the mass of the parcel j is conserved; we do this by assigning a constant mass to each particle, m_j at the outset. This mass is a constant and as long as we keep the same number of particles in our problem (i.e., not destroying any), mass is conserved automatically. We define this mass as $m_j = \rho_j \Delta \upsilon_j$. From this we have a definition for the volume of each particle, m_j/ρ_j. So as the density changes, the volume of the particle changes. We use the conservation of mass equation to determine density as a function of time.

Taking the derivatives, we can rewrite the Eulerian conservation of mass equation in Lagrangian form as

$$\frac{1}{\rho}\frac{d\rho}{dt} = -\nabla \cdot \mathbf{u} \tag{66}$$

To find a form of this equation for use in SPH, we multiply both sides of the equation by the kernel and integrate over the domain υ

$$\int_{\upsilon} W(\mathbf{s}-\mathbf{x},h)\left(\frac{1}{\rho}\frac{d\rho}{dt}\right) d\upsilon = -\int_{\upsilon} W(\mathbf{s}-\mathbf{x},h)\,\nabla \cdot \mathbf{u}\, d\upsilon \tag{67}$$

By the reproducing nature of the kernel (Eqn. 33) we have the left side obtained easily as $\frac{d\rho}{dt}/\rho$ evaluated at position $\mathbf{s}$, and the right side is rewritten using an identity as

$$-\int_{\upsilon} \left[\nabla \cdot \big(W(\mathbf{s}-\mathbf{x},h)\,\mathbf{u}\big) - \mathbf{u}\cdot\nabla W(\mathbf{s}-\mathbf{x},h)\right] d\upsilon \tag{68}$$

where the gradient is applied on the variable $\mathbf{x}$. Next we use Gauss's theorem, which is stated as

$$\int_{\upsilon} \nabla\cdot\mathbf{f}\, d\upsilon = \int_{S} \mathbf{f}\cdot\mathbf{n}\, dS \tag{69}$$

where the last integral is a surface integral over the surface that encloses the volume υ and $\mathbf{n}$ is the outward surface normal.

$$\frac{1}{\rho}\frac{d\rho}{dt} = -\int_{S} W(\mathbf{s}-\mathbf{x},h)\,\mathbf{u}\cdot\mathbf{n}\, dS + \int_{\upsilon} \mathbf{u}\cdot\nabla W(\mathbf{s}-\mathbf{x},h)\, d\upsilon \tag{70}$$

If the surface is far from the point s, then the integrand in the first integral is zero as the weighting function goes to zero rapidly. What remains is the last term.

$$\frac{1}{\rho}\frac{d\rho}{dt} = \int_{\upsilon} \mathbf{u}\cdot\nabla W(\mathbf{s}-\mathbf{x},h)\, d\upsilon \tag{71}$$

The discrete form of this equation for use in SPH, when evaluated at particle x_i, is

$$\Big(\frac{1}{\rho}\frac{d\rho}{dt}\Big)_i = \sum_j \frac{m_j}{\rho_j}\,\mathbf{u}_j\cdot\nabla W_{ij} \tag{72}$$

where $W_{ij} \equiv W(\mathbf{x}_i - \mathbf{x}_j)$. Due to symmetry, the gradient of the weighting function with respect to the second particle, $\nabla_j W_{ij} = -\nabla_i W_{ij}$, the gradient with respect to the first particle in the argument of $W(\mathbf{x}_i - \mathbf{x}_j)$; therefore we can rewrite the above equation for the conservation of mass as

$$\Big(\frac{1}{\rho}\frac{d\rho}{dt}\Big)_i = -\sum_j \frac{m_j}{\rho_j}\,\mathbf{u}_j\cdot\nabla_i W_{ij} \tag{73}$$

Another form of this equation is found by examining the expression for a gradient (Eqn. 64). From that expression, the gradient of a constant is zero, or

$$0 = \sum_j \frac{m_j}{\rho_j}\,\nabla_i W_{ij} \tag{74}$$

Multiplying by $\mathbf{u}_i$, we can then add this zero sum to the previous form of the conservation of mass equation.

$$\Big(\frac{d\rho}{dt}\Big)_i = \rho_i \sum_j \frac{m_j}{\rho_j}\,\big(\mathbf{u}_i - \mathbf{u}_j\big)\cdot\nabla_i W_{ij} \tag{75}$$

Often it is necessary to obtain the pressure at a location not corresponding with a particle. It can be calculated by the usual SPH definition:

$$\rho(\mathbf{s}) = \sum_j m_j W(\mathbf{s} - x_j)$$

which indicates that the density at a point is due to the contribution of the neighboring particles (those within the compact support of the kernel at **s**. Another approach that is more accurate is due to Randles and Libersky (1996):

$$\rho(\mathbf{s}) = \frac{\sum_j m_i\, W(\mathbf{s} - x_j)}{\sum_j \left(\frac{m_j}{\rho_j}\right) W(\mathbf{s} - x_j)} \tag{76}$$

SPH calculations can be tamed by using this equation periodically (we use every 40 iterations) to smooth out density variations (Colagrossi and Landrini (2003), Panizzo, Dalrymple and Rogers (2006).

3.3 Equations of Motion

Starting from the equations:

$$\frac{d\mathbf{u}}{dt} = -\frac{1}{\rho}\nabla p - \mathbf{g} \tag{77}$$

where $\mathbf{g}$ is the acceleration of gravity and p is the fluid pressure. If we multiply through by ρ and multiply both sides by the kernel and integrate over the domain, we have

$$\left(\frac{d\mathbf{u}}{dt}\right)_i = -\frac{1}{\rho_i}\sum_j \frac{m_j}{\rho_j}\, p_j\, \nabla_i W_{ij}\, - \mathbf{g} \tag{78}$$

To this equation, we can subtract the following zero sum:

$$\frac{p_i}{\rho_i}\left(\sum_j \frac{m_j}{\rho_j}\, \nabla_i W_{ij}\right) = 0$$

which gives us the final form

$$\left(\frac{d\mathbf{u}}{dt}\right)_i = -\sum_j m_j \left(\frac{p_i + p_j}{\rho_i \rho_j}\right) \nabla_i W_{ij}\, - \mathbf{g} \tag{79}$$

A second form can be derived with the equality: $p = (\sqrt{p})^2$. Substituting into the equation of motion, we have

$$\frac{d\mathbf{u}}{dt} = -\frac{2}{\rho}\sqrt{p}\,\nabla p - \mathbf{g} \tag{80}$$

Multiplying through by $\rho/(2\sqrt{p}$ gives

$$\frac{\rho}{2\sqrt{p}}\,\frac{d\mathbf{u}}{dt} = -\nabla p - \frac{\rho}{2\sqrt{p}}\,\mathbf{g} \tag{81}$$

Multiplying this equation by the kernel and integrating over the domain, using the Gauss theorem to remove the boundary term, yields

$$\left(\frac{\rho}{2\sqrt{p}}\,\frac{d\mathbf{u}}{dt}\right)_i = -\sum_j \frac{m_j}{\rho_j}\sqrt{p_j}\,\nabla_i W_{ij} - \left(\frac{\rho}{2\sqrt{p}}\right)_i g \tag{82}$$

Finally, dividing by $(\rho/2\sqrt{p})_i$ results in the final equation:

$$\left(\frac{d\mathbf{u}}{dt}\right)_j = -\sum_j 2\, m_j \frac{\sqrt{p_i\, p_j}}{\rho_i\, \rho_j}\, \nabla_i W_{ij} - \mathbf{g} \tag{83}$$

Another form of the equation of motion often used is

$$(\frac{d\mathbf{u}}{dt})_j = -\sum_j m_j (\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2}) \nabla_i W_{ij} - \mathbf{g} \tag{84}$$

3.4 Pressure and Sound Speed

The pressure for each particle is obtained from an equation of state. Monaghan (1994) uses a form due to Batchelor (1973):

$$p = B\rho_s [(\frac{\rho}{\rho_s})^\gamma - 1] \tag{85}$$

The B is a constant, ρ_s is a reference density, usually taken as the density of the fluid at the free surface, so that the pressure is zero there, and γ is the polytropic constant, usually between 1 and 7. A value of 7 fits oceanic data, while Morris *et al.* (1997) suggests for low Reynold number flows that γ is best taken as unity. (The inclusion of the minus one term in the equation of state is to give zero pressure at a surface.)

The advantage of using an equation of state is that there is no need to solve for a partial differential equation for pressure–this equation is usually a Poisson equation, which is time consuming to solve.

The speed of sound is given by the derivative of this equation of state with respect to density:

$$C^2(\rho) = \frac{\partial p}{\partial \rho} = B\gamma(\frac{\rho}{\rho_s})^{\gamma-1} = C_s^2(\frac{\rho}{\rho_s})^{\gamma-1} \tag{86}$$

where B is now seen to correspond to C_s^2/γ, the speed of sound at the surface divided by the polytopic constant, γ.

The choice of B is very important. If we use a value corresponding to the real value of the speed of sound in water, then for numerical modeling, we would have to chose a very small time step, based on such criteria as a Courant-Fredrich-Levy condition. Monaghan has found by experience that the sound speed could be artificially slowed significantly for fluids without affecting the fluid motion; however, he argues that the minimum sound speed should be about ten times greater than the maximum expected flow speeds.

If we examine the hydrostatic pressure law (that is, the static version of our equation of motion, but only in the vertical direction, we have

$$0 = -\frac{1}{\rho}\frac{\partial p}{\partial y} - g = -\frac{1}{\rho}\frac{\partial p}{\partial \rho}\frac{\partial \rho}{\partial y} - g \tag{87}$$

Substituting for $\partial p/\partial \rho$ using our equation of state, yields an equation for the variation of density with elevation.

$$\frac{\partial \rho}{\partial y} = -\rho g/C^2(\rho) \tag{88}$$

Solving by separation of variables, we obtain the hydrostatic variation of the pressure

$$\rho = \rho_s\Big(1 - \Big(\frac{\gamma-1}{\gamma}\Big)\frac{gy}{B}\Big)^{\frac{1}{\gamma-1}} \tag{89}$$

except for γ=1, when the result is $\rho = \rho_s \exp(-gy/C_s^2)$. These two relationships provide a way to initialize the density in an SPH model.

3.5 Time-stepping

There are a variety of ways to march the solution of the SPH equations ahead in time. It is desirable to use at least a second order accurate scheme in time. Runge-Kutta and other methods have been used. To illustrate some of these methods, we first digitize time for the computer: $t = 0, \Delta t, 2\Delta t, \ldots, n\Delta t, (n+1)\Delta t, \ldots$.

Monaghan and colleagues use a modified Euler predictor-corrector method for second order accuracy. The governing equation is taken in this form

$$\frac{d\mathbf{v}}{dt} = \mathbf{f} \tag{90}$$

where we will assume that f involves $\mathbf{v}$ in some way. The first step is to predict at the next time level using the Euler method:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + f_n \, \Delta t \tag{91}$$

The value of $\mathbf{v}_{n+1/2}$ (at the time midpoint) is $(\mathbf{v}_{n+1} + \mathbf{v}_n)/2$, which is used to calculate $f_{n+1/2}$. The correction step is then

$$\mathbf{v}_{n+1} = \mathbf{v}_n + f_{n+1/2} \, \Delta t$$

Monaghan (1989) uses this development to show that the SPH method conserves both linear and angular momentum. In practice, they used the midpoint value of the previous time step instead of computing f_n, which saves time and creates only a small error. The overall scheme is second order.

One of the time-marching methods used in the Johns Hopkins model is the Verlet (1967) or Newmark (1959) method. The Verlet method can be derived from Taylor series in time for the position $\mathbf{x}$:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \mathbf{x}'(t)\Delta t + \mathbf{x}''(t)\Delta t^2/2 + \mathbf{x}'''(t)\Delta t^3/6 + O(\Delta t^4) \tag{92}$$

$$\mathbf{x}(t-\Delta t) = \mathbf{x}(t) - \mathbf{x}'(t)\Delta t + \mathbf{x}''(t)\Delta t^2/2 - \mathbf{x}'''(t)\Delta t^3/6 + O(\Delta t^4) \tag{93}$$

where the primes are derivatives with respect to time. Adding these two equations we obtain a very accurate (fourth order in Δt) way to advance the position of a particle, based on knowing the position of the particle two times steps earlier and the acceleration at time level t:

$$\mathbf{x}(t+\Delta t) = 2\mathbf{x}(t) - \mathbf{x}(t-\Delta t) + a(t)\,\Delta t^2 + O(\Delta t^4) \tag{94}$$

where $a(t) = \mathbf{x}''(t)$. Subtracting the equations instead gives the simple Verlet equation for the velocity

$$v(t) = \frac{\mathbf{x}(t+\Delta t) - \mathbf{x}(t-\Delta t)}{2\Delta t} + O(\Delta t^2), \tag{95}$$

which is two orders in Δt less accurate than the position expression. The so-called Velocity Verlet equation is found by expressing the velocity in a Taylor series:

$$v(t+\Delta t) = v(t) + v'(t)\Delta t + v''(t)\Delta t^2 \ldots = v(t) + a(t)\Delta t + \mathbf{x}'''\Delta t^2 + O(\Delta t^3) \tag{96}$$

We can obtain an expression for $\mathbf{x}'''(t)$ by differentiating the Taylor series for $\mathbf{x}(t+\Delta t)$ twice with respect to time, which gives us $a(t+\Delta t) = a(t) + x'''(t)\Delta t + O(\Delta t^2)$. Solving for $\mathbf{x}'''(t)$ and substituting into our expression above, we obtain

$$v(t+\Delta t) = v(t) + \frac{1}{2}\Delta t\,\big(a(t) + a(t+\Delta t)\big) + O(\Delta t^3) \tag{97}$$

These equations can then be rewritten as:

$$\mathbf{x}_{n+1} = 2\mathbf{x}_n - \mathbf{x}_{n-1} + a_n\,\Delta t^{\,2} \tag{98}$$

$$v_{n+1} = v_n + \frac{1}{2}\Delta t\,(a_n + a_{n+1}) \tag{99}$$

In practice, these equations are used with a half step in time so that the variables at three levels in time $(n+1), n$, and $(n-1)$ are not stored:

$$v_{n+1/2} = v_n + a_n\,\Delta t/2 \tag{100}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + v_{n+1/2}\,\Delta t \tag{101}$$

$$v_{n+1} = v_{n+1/2} + a_{n+1}\,\Delta t/2 \tag{102}$$

The general form for the Newmark method is

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{v}_n\Delta t + \frac{\Delta t^2}{2}\left[(1-2\beta)\,\mathbf{a}_n + 2\beta\,\mathbf{a}_{n+1}\right] \tag{103}$$

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \Delta t\left[(1-\gamma)\,\mathbf{a}_n + \gamma\,\mathbf{a}_{n+1}\right] \tag{104}$$

For $\beta = 0$ and $\gamma = 1/2$, this yields the (velocity) Verlet method; for $\beta = 1/4$ and $\gamma = 1/2$, this reduces to the trapezoidal rule, and for $\beta = 0$ and $\gamma = 1/4$, the central difference method results.

Another method, popular in molecular dynamics, is the Beeman (1976) method, which is stated as

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{v}_n\Delta t + \big(\frac{2}{3}\mathbf{a}_n - \frac{1}{6}\mathbf{a}_{n-1}\big)\Delta t^2 + O(\Delta t^4) \tag{105}$$

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \big(\frac{1}{3}\mathbf{a}_{n+1} + \frac{5}{6}\mathbf{a}_n - \frac{1}{6}\mathbf{a}_{n-1}\big)\Delta t + O(\Delta t^3) \tag{106}$$

which is more accurate for velocity than the Verlet method.

A predictor-corrector version of the Beeman method for velocity that is useful when the acceleration terms are dependent on the velocity and the position uses the same equation as above for position $\mathbf{x}_{n+1}$, but uses the following predictor-corrector equations for velocity.

$$\mathbf{v}_{p,n+1} = \mathbf{v}_n + \big(\frac{3}{2}\mathbf{a}_n - \frac{1}{2}\mathbf{a}_{n-1}\big)\Delta t \tag{107}$$

where everything on the righthand side is known. Then the corrected value for $\mathbf{v}_{n+1}$ is

$$\mathbf{v}_{c,n+1} = \mathbf{v}_n + \big(\frac{1}{3}\mathbf{a}_{n+1} + \frac{5}{6}\mathbf{a}_n - \frac{1}{6}\mathbf{a}_{n-1}\big)\Delta t \tag{108}$$

This gives a scheme that is correct to order Δt^4 for both position and velocity.

The derivation of the Beeman methods comes about by assuming, for example, that the predictor can be described as

$$\mathbf{x}_{p,n+1} = \mathbf{x}_n + \mathbf{v}_n\,\Delta t + \Delta t^{\,2}\sum_{p}^{q-1} b_p\,\mathbf{a}_{n-p+1} \tag{109}$$

where q is the order of the method. The correctors are

$$\mathbf{x}_{c,n+1} = \mathbf{x}_n + \mathbf{v}_n \,\Delta t + \Delta t^2 \sum_p^{q-1} c_p \,\mathbf{a}_{n-p+2} \tag{110}$$

$$\mathbf{v}_{c,n+1} \,\Delta t = \mathbf{x}_{n+1} - \mathbf{x}_n + \Delta t^2 \sum_p^{q-1} d_p \,\mathbf{a}_{n-p+2} \tag{111}$$

Taking $q = 3$ and using Taylor series about time level n for $\mathbf{v}_{n+1}, \mathbf{a}_{n+1}$, and $\mathbf{a}_{n-1}$, results in Eqn. 105 from the first of these two equations. The second equation yields

$$\mathbf{v}_{c,n+1} \,\Delta t = \mathbf{x}_{n+1} - \mathbf{x}_n + \Delta t^2 \big(\frac{2}{3}\,\mathbf{a}_{n+1} + \frac{1}{6}\,\mathbf{a}_n\big) \tag{112}$$

Substituting from the predictor equation for $\mathbf{x}_{n+1} - \mathbf{x}_n$ produces Eqn. 108.

A more accurate form of this equation is found by using the Taylor Series expansion of

$$\mathbf{v}_{c,n+1} = \mathbf{v}_n + \Delta t \sum_p^{q-1} b_p \,\mathbf{a}_{n-p+2} \tag{113}$$

with $q = 4$ leads us to an even more accurate corrector than given by Beeman. We find, through the use of Taylor Series for $\mathbf{v}_{n+1}, \mathbf{a}_{n+1}$, and $\mathbf{a}_{n-1}$,

$$\mathbf{v}_{c,n+1} = \mathbf{v}_n + \Delta t \big(\frac{5}{12}\mathbf{a}_{n+1} + \frac{2}{3}\mathbf{a}_n - \frac{1}{12}\mathbf{a}_{n-1}\big), \tag{114}$$

which is a Adams-Moulton third order method and it doesn't use any more information than was used with Beeman's third order method.

3.5.1 Nearest Neighbors

In principle, the summations in all the particle equations are over the entire number of particles in the problem. However, since the support for the kernel is small, the summations need actually only include those particles in the neighborhood (within a distance of 2 h). This reduction of terms in the summations reduces the computations for SPH by at least an order of magnitude. Further, symmetry provides more time-saving, as the force on a given particle due to the presence of another, is equal and opposite to the force that it exerts on the other particle.

To take advantage of the reduced number of calculations for the summations, we have to keep track of the nearest neighbors of each particle, or at least the particles that exist within a certain neighborhood. One thing that helps us is that the SPH is a Lagrangian method and we know the trajectory of each particle through time from its initial position. To do this, we number the paricles initially and the particles retain the same number throughout the computation.

One of the simplest ways to keep track of neighbors is to periodically (or even each time step) determine which particles are in a rectangular mesh that is spread over the problem, Monaghan and Gingold (1983). This mesh, with a cell size, of say $2h$ in each spatial dimension, extends over the entire domain. Then using the location information $\mathbf{x}$ for each particle, we determine the cell the particle is in. At the end of this procedure, we then know the particles (by their numbers) in each cell; that is to say, we know the nearest neighbors of the particles in the cell. Further, we know the particles in the cells that surround the cell of interest, which are close enough to possibly participate in a summation. In one dimension, the cells to either side of a given cell are important; in 2 D, there are eight other cells that surround the cell of interest, and in 3D, it is 26.

Another method is the *tree search algorithm* In two dimensions, the procedure is described by two basic procedures. The first involves dividing the domain into quadrants. This defines a level. If there is more than one particle in a quadrant, then the quadrant is subdivided into smaller quadrants, defining the next level for that branch. This process continues, defining more levels, until only a single particle falls within a quadrant. This final hierarchical structure is the tree. Then the tree is used to search for the nearest neighbors. Hernquist and Katz (1989)

3.6 Viscosity and Tensile Instability

Without viscosity the SPH method is subject to instabilities–one manifestation is that each of the particles begins moving chaotically–we have called it "boiling." Monaghan and Gingold (1983); Monaghan (2000) introduced an artificial viscosity to simulate a viscosity, dissipate energy within shock fronts, and to prevent particles from interpenetrating other fluid masses.

$$\Pi_{ij} = \begin{cases} -\left(\frac{\alpha \bar{c}_{ij}\phi_{ij} - \beta\phi_{ij}^2}{\bar{\rho}_{ij}}\right), & \mathbf{u}_{ij}\cdot\mathbf{x}_{ij} < 0 \\ 0, & \mathbf{u}_{ij}\cdot\mathbf{x}_{ij} \geq 0 \end{cases} \tag{115}$$

where

$$\phi_{ij} = \frac{h_{ij}\mathbf{u}_{ij}\cdot\mathbf{x}_{ij}}{|\mathbf{x}_{ij}|^2 + \phi^2} \tag{116}$$

$$\bar{c}_{ij} = \frac{1}{2}(c_i + c_j) \tag{117}$$

$$\bar{\rho}_{ij} = \frac{1}{2}(\rho_i + \rho_j) \tag{118}$$

$$h_{ij} = \frac{1}{2}(h_i + h_j) \tag{119}$$

$$\mathbf{u}_{ij} = \mathbf{u}_i - \mathbf{u}_j \tag{120}$$

$$\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j \tag{121}$$

This term is active as particle approach each other and zero otherwise. The constants α and β are on the order of unity.

For low Reynolds number flows, Morris *et al.* (1997) and Morris (2000) use a more familiar form.

$$\frac{1}{\rho}(\nabla\cdot\mu\nabla)\mathbf{u}_i = \sum_j m_j \frac{\nu_o(\rho_i+\rho_j)}{\bar{\rho}_{ij}^2}\,\frac{\mathbf{x}_{ij}\cdot\nabla_i W_{ij}}{x_{ij}^2+\eta^2}\,\mathbf{u}_{ij} \tag{122}$$

Here ν_o is the viscosity, 10^{-6} m^2, and $\eta^2 = 0.01h^2$, a term added to avoid singularities.

Another method introduced by Monaghan for keeping particle motions more regular and from interpenetrating is the XSPH method:

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}_i - \epsilon\sum_j \frac{m_j}{\rho_j}(\mathbf{u}_i - \mathbf{u}_j)\,W_{ij} \tag{123}$$

where ϵ is a constant around 0.3. This method ensures that particles in a neighborhood move with roughly the same velocity.

3.7 Turbulence

The artificial viscosity term becomes very strong with very large numbers of particles and results in too much damping. As a result, Rogers and Dalrymple (2004) developed a compressible SPH approach to subparticle scaling of turbulence. The governing equation is the LES formulation for the equation of motion, which is developed through Favre-averaging ($\tilde{f} = \overline{\rho f}/\bar{\rho}$)

$$\frac{d\tilde{\mathbf{u}}}{dt} = -\frac{1}{\bar{\rho}}\nabla\bar{p} + \frac{1}{\bar{\rho}}\left(\nabla\cdot\bar{\rho}\nu_o\nabla\right)\mathbf{u} + \frac{1}{\bar{\rho}}\nabla\cdot\tilde{\tau} \tag{124}$$

where

$$\tau_{ij} = \bar{\rho}\left(2\nu_t\tilde{S}_{ij} - \frac{2}{3}\tilde{S}_{kk}\delta_{ij}\right) - \frac{2}{3}\,\bar{\rho}\,C_I\bar{\Delta}^2\delta_{ij}$$

The constant C_I is taken as 0.00066 according to Blin *et al.* (2002). The Favre-averaged strain tensor

$$\tilde{S}_{ij} = \frac{1}{2}\Big(\frac{\partial\tilde{u}_i}{\partial x_j} + \frac{\partial\tilde{u}_j}{\partial x_i}\Big)$$

The turbulent viscosity is obtained by a Smagorinsky model $\nu_t = \big(C_s\Delta l\big)|\bar{S}|$, with C_s =0.12. Δl is the initial particle spacing when the model is started and $|\bar{S}|$, the local strain rate, is defined as

$$|\bar{S}| = \left(2\bar{S}_{ij}\bar{S}_{ij}\right)^{1/2}$$

To implement the Fauvre-averaged equations in SPH, we have to put it into particle form. Only the last term is new; it is implemented as

$$\frac{1}{\rho}\nabla\cdot\tau\ |_i = \sum_j m_j\Big(\frac{\tau_i}{\rho_i^2} + \frac{\tau_j}{\rho_j^2}\Big)\cdot\nabla_i W_{ij} \tag{125}$$

3.8 Boundaries

One of the problems with SPH is boundary specification. The domain of interest is usually surrounded by either fixed or moving boundaries, or in the case of water waves, a free surface. For fixed or moving bounding surfaces, the position of the boundary is always known. The boundary condition is the the fluid particles next to the surface cannot pass through the surface. For the fixed surface, this is specified by a no-flow condition: $\mathbf{u}\cdot\mathbf{n} = 0$, where $\mathbf{n}$ is the surface normal. For a moving surface, the velocity of motion normal to the surface must have the same speed as the surface; this is the *kinematic boundary condition.*

There are a variety of ways to implement boundary conditions. The simplest is to simply place particles on the boundary, which do not move. However, in most other ways, the boundary particles are treated in the same way as all the other particles. This presents a bumpy surface as a fluid particle moving parallel to the surface will have a different force acting on it depending on the distance to the nearest boundary particle. Dalrymple and Knio (2001) used a double staggered layer of particles to reduce this effect and to ensure that particles did not penetrate the boundary.

Monaghan and Kos (1999) developed the Lennard-Jones boundary force model. As a particle approaches a wall it experiences a force

$$\mathbf{f} = \mathbf{n}\,R(y)P(x) \tag{126}$$

where $\mathbf{n}$ is the wall normal, $R(y)$ is a repulsion force based on the (perpendicular) distance to the wall (y),

$$R(y) = A\,\frac{1}{\sqrt{q}}\,(1-q)$$

where A is defined as 0.01 C_i^2/h, where C_i is the sound speed at the particle i and q is the normalized distance from the wall, $y/2h$. The function $P(x)$, where x is the distance on the boundary chord linking the two nearest boundary particles, which are separated by a distance Δb, was developed to provide the near-boundary particle with a constant force as it moves parallel to the wall.

$$P(x) = \frac{1}{2}\big(1 + \cos\left(\frac{2\pi x}{\Delta b}\right)\big)$$

3.9 Moving Least Squares, CSPH, and RKPM methods for SPH

During the derivation of the particle forms of the conservation of mass and the equations of motion, the use of Gauss' Divergence Theorem led to surface integrals that were neglected with the argument that the integrands (involving the kernel) decayed rapidly with distance and therefore would yield zero contributions to the integral. So the terms were dropped. However, in the vicinity of boundaries this is not true and corrections need to be made. Belytschko *et al.* (1996), Dilts (1999), Cueto-Felgueroso *et al.* (2004) argues that the Moving Least Squares provides a better consistency for SPH than the current method. Again, MLS uses monomial basis functions to ensure that a local polynomial expansion best fits the surrounding data points. Further there is less oscillation of the interpolating polynomial between particles with the MLS, which Dilts argues is the source of the tensile instability. Others have argued for a MLS approach to SPH: Belytschko *et al.* (1996) In addition, corrections are needed near boundaries, where the boundary integrals were dropped in the derivation of the particle equations (see Eqn. 70).

3.10 Moving Particle Semi-Implicit and Incompressible SPH

Koshizuka *et al.* (1995) developed an offshoot of SPH, called Moving Particle Semi-implicit (MPI) to address incompressible fluids. The method has been used in Japan quite a bit, lead by Gotoh, *e.g.* Gotoh and Sakai (1999). The model solves a Poisson equation for pressure. The kernel used in MPS was given earlier in Eqn. 43. Gotoh *et al.* (2001) developed a sub-particle-scale model for MPS.

An imcompressible SPH was developed by Lo and Shao (2002). The advantage of this model is that the fluid is incompressible as in MPS and the time step is based on the maximum fluid velocity, not the sound speed; the disadvantage is that the pressure has to be numerically obtained, requiring many more calculations. A question that is currently be resolved is whether the larger time steps but more calculations is faster than the SPH with its smaller time steps but with a simple pressure calculation (the equation of state).

3.11 Modeling Waves

Monaghan (1994) showed a simple wave created by the motion of a paddle at one end of a numerical wave tank, along with simulations of dam breaks. Monaghan and Kos (1999) later applied the model to a solitary wave shoaling on a beach. Dalrymple and Rogers (2006), examined in fair detail breaking waves on a beach, examining the generation of downbursting and the breaking process. A wave breaking against a structure was shown by Gómez-Gesteira and Dalrymple (2004).

Breaking water waves entrap air when there is a strong plunging wave or during impact with with a wall. For this reason, modeling both the water and air for breaking waves provides a way to include the cushioning effects of the air. Colagrossi and Landrini (2003) and Cuomo and Dalrymple (2007) have examined the two-phase problem.

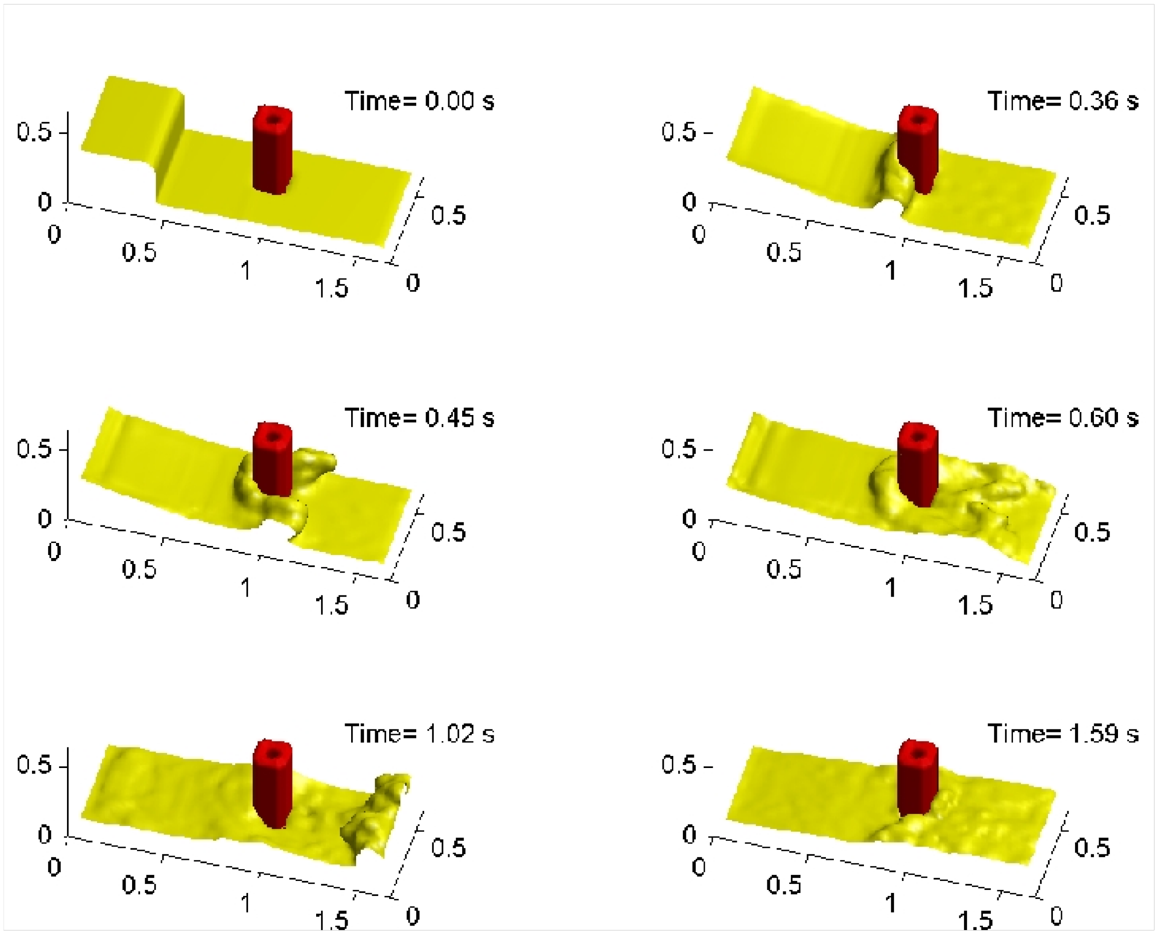

Figure 11: 3D SPH dam break onto a structure (Gomez-Gesteira and Dalrymple, 2004)

3.11.1 Suspended Sediment

The SPH can be used to solve more equations than the hydrodynamic equations for the flow. For example, Zou and Dalrymple examine suspended sediment under waves, using the advection-diffusion equation for the sediment concentration, including a fall velocity. This equation is put into particle form and then solved along with the hydrodynamics equations. By using a pick-up function along the bottom boundary the suspended sediment under waves can be modeled.

References

Airy, G. (1845). *Encyclopaedia Metropolitana*, chapter Tides and Waves.

Batchelor, G. (1973). *An Introduction to Fluid Mechanics*. Cambridge University Press.

Beeman, D. (1976). Some multistep methods for use in molecular dynamics calculations. *Journal of Computational Physics*, **20**, 130–139.

Belytschko, T., Liu, Y., and Gu, L. (1994). Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering*, **37**, 229–256.

Belytschko, T., Krongauz, Y., Fleming, M., Organ, D., and Liu, W. (1996). Smoothing and accelerated compuations in the element free galerkin method. *Journal of Computational and Applied Mathematics*, **74**, 111–126.

Benz, W. (1990). *The Numerical Modelling of Nonlinear Stellar Pulsations*, chapter Smooth Particle Hydrodynamics: A Review, pages 269–288. Kluwer Academic Publishers.

Blin, L., Hadjadj, A., and Vervisch, L. (2002). Large eddy simulation of turbulent flows in reversing systems. In P. Vuillermoz, P. Comte, and M. Lesieur, editors, *Selected Proceedings of the 1st French Seminar on Turbulence and Space Launchers*.

Boussinesq, J. (1872). Theorie des ondes et des remous qui se propagent le long d'un canal rectangulaire horizontal. *Journal de Mathematiques Pures et Appliques*, **17**, 55–108.

Buhman, M. (2000). A new class of radial basis functions with compact support. *Mathematics of Computation*, **70**(233), 307–318.

Colagrossi, A. and Landrini, M. (2003). Numerical simulation of interfacial flows by smoothed particle hydrodynamics. *Journal of Computational Physics*, **191**(2), 448–475.

Craik, A. (2004). The origins of water wave theory. *Annual Review of Fluid Mechanics*, **36**, 1–28.

Cueto-Felgueroso, L., Colominas, I., Mosqueira, G., Navarrina, F., and Casteleiro, M. (2004). On the Galerkin formulation of the smoothed particle hydrodynamics method. *International Journal for Numerical Methods in Engineering*, **60**, 1475–1512.

Cuomo, G. and Dalrymple, R. (2007). A compressible two-phase SPH-LES model for interfacial flows with fragmentation. In preparation.

Dalrymple, R. and Knio, O. (2001). SPH modelling of waves. In ASCE, editor, *Proc. Coastal Dynamics, Lund, Sweden*, pages 779–787.

Dalrymple, R. and Rogers, B. (2006). Numerical modeling of water waves with the SPH method. *Coastal Engineering*, **53**(2/3), 141–147.

Dean, R. (1965). Stream function representation of nonlinear ocean waves. *Journal of Geophysical Research*, **70**(18), 4561–4572.

Dean, R. and Dalrymple, R. (1991). *Water Wave Mechanics for Engineers and Scientists*. World Scientific.

Dilts, G. (1999). Moving-least-square-particle hydrodynamics-I. consistency and stability. *International Journal for Numerical Methods in Engineering*, **44**, 1115–1155.

Gingold, R. and Monaghan, J. (1977). Smoothed Particle Hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, **181**, 375–389.

Gómez-Gesteira, R. and Dalrymple, R. (2004). Using a 3d SPH method for wave impact on a tall structure. *Journal of Waterway, Port, Coastal, and Ocean Engineering*, **130**(2), 63–69.

Gotoh, H. and Sakai, T. (1999). Lagrangian simulation of breaking waves using particle method. *Coastal Engineering Journal*, **41**(3/4), 303–326.

Gotoh, H., Shibihara, T., and Hayashi, M. (2001). Sub-particle-scale model for the mps method-lagrangian flow model for hydraulic engineering. *Computational Fluid Dynamics Journal*, **9**(4), 339–247.

Hardy, R. (1971). Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, **176**, 1905–1915.

Hardy, R. (1990). Theory and application of the multiquadric-biharmonic method: 20 years of discovery. *Computers and Mathematics with Applications*, **19**(8/9), 163–208.

Hernquist, L. and Katz, N. (1989). TreeSPH-a unification of SPH with the hierarchical tree method. *The Astrophysical Journal Supplement Series*, **70**, 419–446.

Johnson, G., Stryk, R., and Beissel, S. (1996). SPH for high velocity impact calculations. *Computer Methods in Applied Mechanics and Engineering*, **139**, 347–373.

Kansa, E. (1990). Multiquadrics-a scattered data approximation scheme with applications to computational fluid dynamics-I. *Computers and Mathematics with Applications*, **19**(8/9), 127–145.

Kansa, E. (1999). Motivation for using radial basis functions to solve pdes.

Koshizuka, S., Tamako, H., and Oka, Y. (1995). A particle method for incompressible viscous flow with fluid fragmentation. *Computational Fluid Dynamics Journal*, **4**(1), 29–46.

Lancaster, P. and Salkauskas, K. (1981). Surfaces generated by moving least squares method. *Mathematics of Computation*, **37**(155), 141–158.

Li, S. and Liu, W. (2004). *Meshfree Particle Methods*. Springer.

Liu, G. and Liu, M. (2003). *Smoothed Particle Hydrodynamics: a meshfree particle method*. World Scientific.

Lo, E. and Shao, S. (2002). Simulation of near-shore solitary wave mechanics by an incompressible SPH method. *Applied Ocean Research*, **24**, 275–286.

Lucy, L. (1977). A numerical approach to testing of the fusion process. *Astronomical Journal*, **88**, 1013–1024.

Monaghan, J. (1989). On the problem of penetration in particle methods. *Journal of Computational Physics*, **82**, 1–15.

Monaghan, J. (1992). Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, **30**, 543–574.

Monaghan, J. (1994). Simulating free surface flows with SPH. *Journal of Computational Physics*, **110**, 399–406.

Monaghan, J. (2000). SPH without a tensile instability. *Journal of Computational Physics*, **159**, 2900–311.

Monaghan, J. and Gingold, R. (1983). Shock simulation by the particle method SPH. *Journal of Computational Physics*, **52**, 374–389.

Monaghan, J. and Kos, A. (1999). Solitary waves on a Cretan beach. *Journal of Waterway, Port, Coastal, and Ocean Engineering*, **125**(3), 145–154.

Morris, J. (2000). Simulating surface tension with Smoothed Particle Hydrodynamics. *International Journal for Numerical Methods in Engineering*, **33**(3), 333–353.

Morris, J., Fox, P., and Zhu, Y. (1997). Modeling low Reynolds number incompressible flows using SPH. *Journal of Computational Physics*, **136**, 214–226.

Morse, B., Yoo, T., Rheingans, P., Chen, D., and Subramanian, K. (2001). Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *Proc. Intl. Conf. on Shape Modeling and Application*, pages 89–98. IEEE Computer Society.

Newmark, N. M. (1959). A method of computation for structural dynamics. *J. Eng. Mech.*, **85**(EM3), 67–94.

Randles, P. and Libersky, L. (1996). Smoothed Particle Hydrodynamics: some recent improvements and applications. *Computer Methods in Applied Mechanics and Engineering*, **138**, 375–408.

Rogers, B. and Dalrymple, R. (2004). SPH modeling of breaking waves. In *Proc. 29th International Conference on Coastal Engineering*, pages 415–427.

Stokes, G. (1847). On the theory of oscillatory waves. *Transaction of the Cambridge Philosophical Society*, **8**, 441–455.

Verlet, L. (1967). Computer 'experiments' on classical fluids, I. thermodynamical properties of Lennard-Jones molecules. *Physical Review D*, **159**, 98–103.

Wendland, H. (2005). Computational aspects of radial basis function approximation. In K. Jetter *et al.*, editors, *Topics in Multivariate Approximation and Interpolation*. Elsevier.