Chapter 7

Finite Pointset Method and Weight Functions

7.1 Introduction

Finite Pointset Method is meshless method for solving partial differential equations. It is based on least square approximation or a moving least square approximation. It is fully Lagrangian method to handle problems in fluid dynamics in flow simulation with complicated geometry as well as rapidly changing geometry involving free surface or phase boundaries. In this chapter, we have made an attempt to find the influence of different weight functions on Finite Pointset discretization in terms of convergence and stability.

7.2 Introduction to Meshfree Methods

Meshfree methods use a set of nodes scattered within the problem domain as well as sets of nodes scattered on the boundaries of the domain to represent the problem domain and its boundaries. The increasing complexity of real life problems leads towards the development of new methods. The Finite Element Method has better flexibility, effectiveness, and accuracy in problems involving complex geometry in compare to Finite Difference Method. Now, the limitations of FEM are becoming increasingly evident. For example, large deformations can deteriorate the accuracy because of element distortion. To answer this question, a new and more powerful class of techniques known as Meshfree or Meshless or gridfree methods are emerging.

The classical meshfree Lagrangian method to handle problems in fluid dynamics is the Smoothed Particle Hydrodynamics (SPH). The advantages of such a Lagrangian method come into play when we would like to study time dependent fluid flow process in very complex geometrical structures, structures which are rapidly changing in time as well as process characterized by various phases or by free surfaces. In such cases, mesh based numerical scheme have certain disadvantages since they would have to perform a very time consuming mesh generation as well as possibly a re-meshing procedure practically after each time step. In contrast, a Lagrangian particle method neither needs generation nor re-meshing procedures in principle. SPH was initially developed to study phenomena in astrophysics (see [GM77], [Luc77]). Later, it was developed for flow cases even on earth (see [MFZ97], [CR99], [Mon94], [Mor00]). Unfortunately, SPH has poor approximation properties, especially of the second derivatives, required to model the Navier-Stokes equations. Moreover, it is difficult to incorporate boundary conditions of certain types. In SPH, incompressible flows are approximated by using the compressible approach together with very stiff equation of state.

Smoothed Particle Hydrodynamics(SPHM) is referred to as the first meshfree method. The basic steps of the scheme are as follows: the conservation laws are expressed in the Lagrangian form for primitive variables and the spatial derivatives are approximated; then, the partial differential equations reduced to a time dependent system of ODEs and finally, it is solved by ODE's solver.

This is a grid free method, where the spatial derivatives of a function at a point is approximated by discrete values over a set of neighboring points. These neighboring points are the so called particles and their distribution need not be uniform or regular. Therefore, this method is suitable for fluid dynamical problems with moving boundaries and free surface flows.

In its original formulation, SPH is easy for implementation, but it provides poor accuracy or convergence (see [DO96], [GS01]). Further, variety of meshfree methods were proposed in the last decade for solving fluid dynamics and solid mechanics problems. Detailed discussion on the development of meshfree methods and their applications can be found in (see [AL01], [BKO+96], [DO96], [GS01], [LL], [LDT96], [PY01]). It should be noted that the terminology in this area is still not well established. We have used the term meshfree methods, but one can find in literature different names: particle methods, meshless methods, gridfree methods, gridless methods or clouds methods. These names are used as notation for the group including methods such as SPH, reproducing kernel particle method (RKPM), element free Galerkin methods (EFG), diffuse element methods (DEM), finite cloud methods, generalized finite difference methods (GFDM), etc.. Some of these methods are very close or equivalent to each other

and great part of them allow a general consideration as partition unity method (PUM).

Since the classical SPH approach is based on the integral interpolant with sufficiently smoothing kernel function, it does not give good approximation of derivatives of a function near boundaries. The least squares method is an alternative approach to approximate spatial derivatives in a grid free structures (see [BKO+96], [Dil96], [DK98], [Kuh99]). In [Kuh99], it is shown that the moving least square methods gives a better approximation of function and its derivatives near boundaries.

Both of the approaches are similar to the finite difference discretization and show the well known problem of instability. In order to stabilize the scheme, some sort of viscosity should be introduced. In [MG83], an artificial viscosity is introduced in the momentum and energy equations for inviscid flows. Similarly, an artificial viscosity is proposed in [Kuh99] in all equations of the system of ODEs resulting approximation of the space derivatives of Euler equations. The artificial viscosity is used in [MG83] and [Kuh99] to stabilize the numerical scheme.

Fluid dynamics equations with viscosity, e. g. the Navier-Stokes equations, can not be solved appropriately with these artificial viscosities. In order to solve the Navier-Stokes equations, one needs to approximate the first and second order spatial derivatives. The classical SPH approach as well as moving least square approach do not give good approximation of derivatives.

In [Tiw00], the authors have used SPH scheme based on weighted least square approach. In this scheme, they approximate the first and second derivatives by a weighted least squares method. This approach is similar to the moving least squares approach used in [BKO+96] and [Kuh99]. In [Kuh99], the boundary conditions are well treated by the moving least square technique. Like in [Kuh99], authors have replaced the boundary by particles and prescribe the boundary condition on the boundary particles. In [Tiw00], authors have consider the full system of Navier-Stokes equations and the solution of compressible Euler equations are obtained by letting the viscosity and the heat conductivity tend to zero. The scheme is tested for $1D$ shock tube problem considered by Sod (see [Sod78]). The scheme is stable and the numerical solution converge to the exact solutions of the Euler's equations when the number of particles tends to infinity and the viscosity and the heat conduction coefficient tend to zero.

Numerical simulations of free surface flows have many industrial applications like casting, tank filling and others. Many methods have been developed to simulate free surface flows (see [HW65], [KP97], [MPR99]). In [TK03] and in [TM03], author have used the same weighted least square approach to simulate the incompressible free surface flows as a limit of the compressible, viscous Navier-Stokes equations with the equation of state such that flow is weakly incompressible. This

type of equation of state was first used by Monaghan (see [Mon94]) to simulate incompressible free surface flows by SPH. Such an equation of state in the framework of SPH has been further used to simulate incompressible viscous flows in [MFZ97].

The solution of Poisson equation is necessary for instationary problems in incompressible fluid flows. Some authors have considered some projection methods for the Navier-Stokes equations, where the Poisson equation for the pressure has to be solved (see [Cho92]). Several authors have considered the projection method on grid based structure such that Poisson equation can be solved by standard methods like finite element or the finite difference method. The grid based method can be quite complicated if the computational domain change in time or takes complicated shapes. In this case, re-meshing is required and more computational effort is needed. Therefore, a grid free method has certainly advantages in such cases.

A grid free method for solving Poisson equation is given in [TK01]. The main idea in [TK01] is to solve the Poisson equation in a grid free structure such that it can be used in Lagrangian particle projection methods for incompressible flows. The method given in [TK01] is a local iteration process. It is based on the least squares approximation. A function and its derivatives can be approximated by the least squares at an arbitrary point from its discrete values belonging to the surrounding cloud of points. However, the values of a function on the particle position is not given. Only the Poisson equation is given. Therefore, we prescribe an initial guess for the values of a function on each particle position. In every iteration step, we enforce the Poisson equation to be satisfied on each particle in the least squares ansatz.

The boundary conditions can easily be handled. Boundaries can be replaced by a discrete set of boundary particles. For the Dirichlet boundary condition, boundary values are assigned in every iteration step on boundary particles. For the Neumann boundary condition, we again enforce it to be satisfied in the least squares ansatz. Therefore, we add one additional equation in the least squares approximation. The method is stable and the numerical solution converges to a unique fixed points as the number of iteration steps tends to infinity. The method can be applied to coarser as well as finer distribution of points. The convergence rate is slower on finer distribution of points.

7.2.1 Introduction to Finite Pointset Method

A Finite Pointset Method (FPM) is a meshfree method to solve partial differential equations. The computational domain is represented by a finite number of particles (pointset), also referred to as numerical points. These points can be arbitrarily distributed, however, they have to pro-

vide a neighboring relationship governed by the smoothing length, i.e. each point needs to find sufficiently many neighbor points within a ball of certain radius. Considering the equation of fluid dynamics, the numerical points move with fluid velocity and carry all information which completely describes the flow problem concerned. Of course, this is a fully Lagrangian method being appropriate for flow simulations with complicated as well as rapidly changing geometry (see [KTU00]), involving free surfaces (see [TJ02], [TK03]), or phase boundaries (see [HJKT03]).

The FPM is based on least squares approximations, where the higher order derivatives can be approximated very accurately and the boundary conditions can be treated in classical sense (see [Kuh99]). Several computation of flow problems using the method of least squares or moving least squares are reported by different authors (see [Dil96], [Kuh99], [Kuh02], [TK01], [TK02], [TJ02], [TK03], [TM03], [Tiw00]) and other references therein.

In [TM03] and [TK03], the authors have performed simulations of incompressible flows as the limit of the compressible Navier-Stokes equations with the quasi compressible equation of state. This approach was first used in [Mon92] to simulate free surface flows by SPH. The incompressible limit is obtained by choosing a very large speed of sound in equation of state such that the Mach number is of order ≈ 0.1. However, the large value of the speed of the sound restricts the time step to be very small due to the CFL condition.

The Chorin's projection method (see [Cho92]) is a widely used approach to solve the incompressible Navier-Stokes equation in grid based structure. In [TK02], the authors have extended Chorin's projection method to meshfree framework with the help of the weighted least squares method. The Poisson pressure equation is solved by meshfree method. In [TK01], it has been shown that Poisson equation can be solved accurately by this approach for any kind of boundary conditions. The Poisson solver can be adopted in the least square approximation procedure with the condition that the Poisson equation and the boundary condition must be satisfied on each particle. This is a local iterative procedure.

In [TK02], the authors have tested the scheme for Channel flows and driven cavity flows. In [TK02], the authors have performed simulations of steady as well as unsteady flows. In the case of channel flows, the numerical results are compared with exact solutions. In the case of driven cavity flows, the numerical solution is compared with the one obtained from the finite element method. It is found that proposed scheme gives accurate results.

In [TJ02], the author have extended the scheme, presented in [TK02] for free surface flows. Numerical experiments are obtained with and without surface tension forces. The broken dam problem is solved without surface tension forces. The Laplace law (see [LL59]) has been tested

for different shapes of bubbles and the scheme produce Laplace law exactly. Finally, they have shown that the binary drop collision of liquid drops shows that the scheme is suitable for simulation of free surface flows.

The numerical scheme for incompressible and slightly compressible flow phenomena, presented in [TK], is based on the classical projection idea of Chorin (see [Cho92], [TK02]). Due to that, the solutions of Poisson as well as Helmholtz differential equations, in particular, form a central task of FPM. These equations can be solved directly in the given meshfree structure with Dirichlet, Neumann or Cauchy boundary conditions in a very accurate way (see [TK01]).

For some industrial applications, such as simulations of car tank refuelling, several fluid phases like fuel, air and foam might involved. Not all phases can be assumed to be incompressible, as for instance the air inside the tank might be compressed during the filling process. This is rather slow compression with the Mach number tending to zero. However, the compression plays a big role as it partially governs the filling process. Thus, in [TK], the authors have incorporated compressibility effects into the classical re-projection idea and finally come up with an implicit scheme for compressible as well as incompressible flows. Therefore, the authors in [TK] present an idea to simulate low Mach number and incompressible flows with exactly the same procedure, i.e., the incompressible case turns out to be a special case of the compressible regime. They have consider the Navier-Stokes equation as the mathematical model. They have solved these equations by the projection method implicitly. The implicit scheme results in linear second order partial differential equations (Poisson, Helmholtz). The authors in [TK] solve them using the constraint least squares method suggested in [TK01] as well as in [TK].

Most of the methods for solving multiphase flows are based on mesh grid techniques (see [BKZ92], [GW01], [HW65], [HN81], [KP97]), where additional computational effort has to be put in order to model the dynamics of the interphase boundaries. The advantage of using the particle method is that phases can be distinguished by simply assigning flags to the fluid particles which identify their proper phase. The phase flags are carried in the same fashion as all other physical data. Since the particle move with fluid velocity they may scatter or accumulate together. If they scatter and create holes in the computational domain, singularities may arise. Hence, holes have to be detected and new particles have to be added. Similarly, any two particles being too close to each other, have to be replace by a single one.

In [TK], the author have excluded surface tension effects. The CSF model (see [BKZ92]) can be easily extended by using the approach proposed by the authors in (see [Mor00]). In [TK], authors have obtained results from convergence studies for general second order linear partial differential equations. If the coefficients are constant, the scheme has second order convergence.

If the coefficients are discontinuous which occur for solving multi-phase flows, the proposed scheme is of first order convergence. The implicit projection method is tested for compressible flows by solving a $1D$ shock tube problem and the results are compared with the exact solutions in [TK]. They have presented two phase flows case for cavity filling, where the air is considered to be compressible.

In [IT02], author has discussed two generalized (meshfree) finite difference methods (GFDM) for the Poisson equation. These are methods due to Liszka and Orkisz (see [LO80]) and to Tiwari (see [TK01]). Both methods are based on moving least squares (MLS) approach for deriving the discretization. The relative comparison shows that the second method is preferable because it is less sensitive to topological restrictions on the nodes distribution. An extension of the second method is presented, which allows for accounting for internal interfaces, associated with discontinuous coefficients. The numerical experiments illustrate the second order convergence of the proposed GFDM for interface problems.

Let us consider problems in gas dynamics governed by the Euler equations

$$\mathbf{I}\cdot\frac{d\mathbf{v}}{dt}+\mathbf{A}\cdot\frac{\partial\mathbf{v}}{dx^{(1)}}+\mathbf{B}\cdot\frac{\partial\mathbf{v}}{dx^{(2)}}+\mathbf{C}\cdot\frac{\partial\mathbf{v}}{dx^{(3)}}=S \tag{7.1}$$

where

$$\mathbf{V}=(\rho,v^{(1)},v^{(2)},v^{(3)},p)^T$$

$$S=\text{source terms}$$

$$\mathbf{A}=\begin{pmatrix}0&\rho&0&0&0\\0&0&0&0&\frac{1}{\rho}\\0&0&0&0&0\\0&0&0&0&0\\0&\rho c^2&0&0&0\end{pmatrix}$$

$$\mathbf{B}=\begin{pmatrix}0&0&\rho&0&0\\0&0&0&0&0\\0&0&0&0&\frac{1}{\rho}\\0&0&0&0&0\\0&0&\rho c^2&0&0\end{pmatrix}$$

$$\mathbf{C}=\begin{pmatrix}0&0&0&\rho&0\\0&0&0&0&\frac{1}{\rho}\\0&0&0&0&0\\0&0&0&0&\frac{1}{\rho}\\0&0&0&\rho c^2&0\end{pmatrix}$$

As we know that the classical SPH method discretizes the system of equations (7.1) in Lagrangian form, i.e. they are based on particles suppose to act as carriers of mass, momentum and energy; the particles move with fluid velocity. It is also possible to incorporate some sort of heat conduction into particle scheme, where viscous terms are treated in a simple way. The approximation strategy used in SPH is based on weighting kernels as basis functions in order to handle occurring spatial derivatives. There are two major disadvantage of the classical SPH methods. These are in fact:

- Difficulties when incorporating boundary condition in to the scheme,
- Necessity of employing artificial viscosity terms in order to keep computation stable.

In [Kuh02], the author concentrates on particle upwind scheme where the employment of artificial viscosity is avoided. They have shown how to treat boundary particles or more generally how to treat particles that might not move with fluid velocity, i.e. non-Lagrangian particles.

Meshfree techniques play an important increasing role as solution methods for conservation laws. Practically, all meshfree methods are based on cloud of points, where each point carries the relevant information for the problem to be solved. One of the biggest advantages of this method is that no mesh has to be established. Generating a grid can be very costly, sometimes it is even the dominating part of the problem. Compared to that, it is relatively simple to establish a cloud point, even within very complex geometries. Moreover, the cloud is easy to maintain or to adapt locally. Due to the free movement of the points, an optimal adaptivity of the cloud is provided towards change in the geometry or towards movement of free surface as well as phase boundaries.

Among the pioneering meshfree methods, Smoothed Particle Hydrodynamics is certainly the most famous (see [Mon92]). SPH is a Lagrangian idea, which is based on the movement of finite mass points. However, for a long time, SPH was suffering from several problems, among them stability and consistency are main. Facing these problems, the development of meshfree methods went into various directions, starting in early nineties. On one hand, people tried to improve SPH. One idea was to avoid inconsistency problems of SPH by reproducing Kernel methods (see [ZLJ95]), another idea was to improve the approximation properties of SPH by the introduction of so called Moving Least Squares (MLS) ideas (see [Dil96], [Kuh99]). On the other hand, many new type of meshfree methods were developed. Widely used methods are Element Free Galerkin (EFG) ideas (see [LBG94]) or the Partition of Unity method (PUM) (see [MB97]). As the EFG and PUM ideas provide the possibility to carry out Finite Element Computations on grid free structures, in [HJKT03] the author introduces meshfree Finite volume and Finite

difference concepts.

In [HJKT03], the author presents the Finite volume particle method (FVPM) which incorporates finite volume ideas into a meshfree framework (see [HJSS00]). In particular, the approach uses the concept of numerical flux functions and guarantees conservation on a discrete level. They have also presented the Finite Pointset Method (FPM) which is a general finite difference method for conservation laws on a meshfree basis (see [Kuh99], [Kuh02]). The latest FPM development which covers the discretization of the incompressible Navier-Stokes equations with multiple phases is presented in [HJKT03].

We have presented the literature survey on SPH and FPM for flow problems which includes regular geometry or irregular geometry. The problem is still remain unsolved regarding the enhancement of speed of FPM simulations. Therefore, in this chapter, we have made an attempt to make the FPM discretized system of Poisson equation as well as Helmholtz equation. Finally, we have presented the numerical examples which shows the analysis of weight functions system on convergence rate of FPM method.

Now, we shall explain the Least Square Method in detail.

7.3 Least Square Method for Approximation of Derivatives

Let $\psi : \Omega \rightarrow R$ be a scalar function and ψ_i be its discrete values at the particle positions $\mathbf{x}_i$ for $i = 1, 2, \cdots, N$. Consider the problem to approximate spatial derivatives of that particular function $\psi(\mathbf{x})$ at some particle position $\mathbf{x}$ based on discrete values of its neighboring points. In order to restrict the number of points, we introduce the weight function $W = W(\mathbf{x}_i - \mathbf{x}; h)$ with small compact support, where h determine the size of compact support. The weight function can be quite arbitrary, however, it makes sense to choose Gaussian weight function of the form

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} exp(-\alpha \frac{\|\mathbf{x}_i - \mathbf{x}\|^2}{h^2}) & \text{if } \frac{\|\mathbf{x}_i - \mathbf{x}\|}{h} \leq 1; \\ 0 & \text{else.} \end{cases}$$

where α is a positive constant and is considered in the range of 6. So far in our implementation, we allow user given h as a function in space and time. However, no adaptive choice of h is realized yet. Working with user given h implies that new particles will have to be brought into play as the particle distribution becomes too sparse or logically particles will have to be removed from the computation as they become too dense.

Let

$$P(\mathbf{x}, h) = \{\mathbf{x}_i : i = 1, 2, 3, ..., m\}$$

be the set of m neighboring particles $\mathbf{x} = (x, y, z)$ in a ball of radius h. We note that central particle x is one element of the neighbor pointset $P(\mathbf{x}, h)$. For consistency reasons, some obvious restriction are required, for example, in $2D$ there should be at least 5 particle in in addition to the central point and they should be neither on the same line nor on the same circle.

We determine the derivatives of a function by using the Taylor's series expansion and the least square approximation. Hence, consider m Taylor expansion of $\psi(\mathbf{x}_i)$ about $\mathbf{x}$

$$\psi(\mathbf{x}_i) = \psi(\mathbf{x}) + \sum_{j=1}^{m} \frac{\partial \psi \mid j \mid}{\partial x^{j_1} \partial y^{j_2} \partial z^{j_3}} \frac{1}{j!} (x_i - x)^{j_1} (y_i - y)^{j_2} (z_i - z)^{j_3} + \mathbf{e}_i \tag{7.2}$$

for $i = 1, ..., m$, where $\mathbf{e}_i$ is the error in Taylors expansion at the point $\mathbf{x}_i$. Denote the coefficients by

$$a_1 = \frac{\partial \psi}{\partial x}, a_2 = \frac{\partial \psi}{\partial y}, a_3 = \frac{\partial \psi}{\partial z},$$

$$a_4 = \frac{\partial^2 \psi}{\partial x^2}, a_5 = \frac{\partial^2 \psi}{\partial x \partial y}, a_6 = \frac{\partial^2 \psi}{\partial x \partial z},$$

$$a_7 = \frac{\partial^2 \psi}{\partial y^2}, a_8 = \frac{\partial^2 \psi}{\partial y \partial z}, a_9 = \frac{\partial^2 \psi}{\partial z^2}.$$

Let us assume that $\psi(\mathbf{x}) = \psi$ is a known discrete function value at the particle position $\mathbf{x}$. For $m > 9$, this system is over-determined with respect to the unknowns a_i and can be written as

$$\mathbf{e} = M\mathbf{a} - \mathbf{b} \tag{7.3}$$

where

$$M = \begin{pmatrix} dx_1 & dy_1 & dz_1 & \frac{1}{2}dx_1^2 & dx_1 dy_1 & dx_1 dz_1 & \frac{1}{2}dy_1^2 & dy_1 dz_1 & \frac{1}{2}dz_1^2 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ dx_m & dy_m & dz_m & \frac{1}{2}dx_m^2 & dx_m dy_m & dx_m dz_m & \frac{1}{2}dy_m^2 & dy_m dz_m & \frac{1}{2}dz_m^2 \end{pmatrix}$$

$$\mathbf{a} = (a_1, a_2, ..., a_9)^T,$$

$$\mathbf{b} = (\psi_1 - \psi, ..., \psi_m - \psi)^T,$$

$$\mathbf{e} = (e_1, b_2, ..., e_m)^T$$

and

$$dx_i = (x_i - x), dy_i = (y_i - y), dz_i = (z_i - z).$$

The unknowns a_i are computed by minimizing a weighted error over the neighboring points. Thus, we have to minimize the following quadratic function

$$J = \sum_{i=1}^{m} w_i e_i^2 = (M\mathbf{a} - \mathbf{b})^T W (M\mathbf{a} - \mathbf{b}) \tag{7.4}$$

with

$$W = diag(w_1, w_2, w_3, ..., w_m),$$

where

$$w_i = w(\mathbf{x}_i - \mathbf{x}; h).$$

The minimization of J with respect to $\mathbf{a}$ formally yields (if $M^T W M$ is nonsingular)

$$\mathbf{a} = \left(M^T W M\right)^{-1}) \left(M^T W\right)\mathbf{b} \tag{7.5}$$

7.4 Least Square Method for Solving Elliptic Equation

We now consider the following linear, second order, elliptic partial differential equation which represents all equations in the above presented projection scheme

$$A\psi + \mathbf{B} \cdot \nabla\psi + C\Delta\psi = f \tag{7.6}$$

where the coefficients A, $\mathbf{B}$ and C are given and real. We solve this equation either with Dirichlet boundary condition

$$\psi = \phi \tag{7.7}$$

or with Neumann boundary conditions

$$\frac{\partial\psi}{\partial\mathbf{n}} = \phi \qquad on \qquad \Gamma \tag{7.8}$$

In the following, we demonstrate the method to solve (7.6), with either of the conditions given in equation (7.7) and (7.8). To our knowledge, there are two types of methods for directly solving elliptic equations in a given meshfree configuration. The first one is presented in [LO80], which can be directly derived from the equation (7.6). The second one is presented in [TK01], where equations (7.7) and (7.8) are added as constraints in the least squares approximation. The comparisons of both methods are presented in [IT02]. It is found that method presented in [TK01] is more stable and the Neumann boundary condition can be easily induced in the approximation.

We consider $\mathbf{x}$ as a central particle and its set of neighbors

$$P(\mathbf{x}, h) = \{\mathbf{x}_i : i = 1, 2, 3, ..., m\}.$$

Furthermore, we consider the above Taylor's expansion (7.2). In (7.2), we have assumed that $\psi(x) = \psi$ is a known discrete function value at x. Now, let us assume that ψ is not known and denote it by a_0.

We add equations (7.6) and (7.8) as constraints into the m Taylor's expansion (7.2). These two additional equations is rewritten in the following forms:

$$Aa_0 + B_1a_1 + B_2a_2 + B_3a_3 + C\left(a_4 + a_7 + a_9\right) = f \tag{7.9}$$

$$n_1a_1 + n_2a_2 + n_3a_3 = \phi \tag{7.10}$$

where

$$\mathbf{B} = (B_1, B_2, B_3), \quad \mathbf{n} = (n_1, n_2, n_3).$$

Note that, for the Dirichlet boundary condition, we have only $m+1$ equations where we directly prescribe the boundary conditions on the boundary particles. The system hence can be written as

$$\tilde{\mathbf{e}} = \tilde{M}\tilde{\mathbf{a}} - \tilde{\mathbf{b}} \tag{7.11}$$

The Matrix $\tilde{M}$ and the vectors $\tilde{\mathbf{a}}, \tilde{\mathbf{b}}, \tilde{\mathbf{e}}$ are slightly different above and are given by

$$\tilde{M} = \begin{pmatrix} 1 & dx_1 & dy_1 & dz_1 & \frac{1}{2}dx_1^2 & dx_1dy_1 & dx_1dz_1 & \frac{1}{2}dy_1^2 & dy_1dz_1 & \frac{1}{2}dz_1^2 \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . \\ 1 & dx_m & dy_m & dz_m & \frac{1}{2}dx_m^2 & dx_mdy_m & dx_mdz_m & \frac{1}{2}dy_m^2 & dy_mdz_m & \frac{1}{2}dz_m^2 \\ A & B_1 & B_2 & B_3 & C & 0 & 0 & C & 0 & C \\ 0 & n_1 & n_2 & n_3 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\tilde{\mathbf{a}} = (a_0, a_1, a_2, ..., a_9)^T,$$

$$\tilde{\mathbf{b}} = (\psi_1, ..., \psi_m, f, \phi)^T,$$

$$\tilde{\mathbf{e}} = (e_1, e_2, ..., e_m, e_{m+1}, e_{m+2})^T.$$

Now, we minimize the functional

$$\tilde{J} = \sum_{i=1}^{m+2} = w_i\tilde{e}_i^2 \tag{7.12}$$

where

$$e_{m+1} = (A\psi + \mathbf{B}\cdot\nabla\psi + C\Delta\psi - f)\,,$$
$$e_{m+2} = \left(\frac{\partial\psi}{\partial\mathbf{n}} - f\right)$$

and

$$w_{m+1} = w_{m+2} = 1.$$

Similarly, the minimization of $\tilde{J}$ with respect to $\mathbf{a}$ formally yields (if $\tilde{M}^T\tilde{W}\tilde{M}$ is nonsingular)

$$\tilde{\mathbf{a}} = \left(\tilde{M}^T\tilde{W}\tilde{M}\right)\left(\tilde{M}^T\tilde{W}\right)\tilde{\mathbf{b}} \tag{7.13}$$

with

$$\tilde{w} = diag(w_1, w_2, ..., w_m, 1, 1).$$

The vector $(\tilde{M}^T\tilde{W})\tilde{\mathbf{b}}$ is explicitly given by

$$(\tilde{M}^T\tilde{W})\tilde{\mathbf{b}} = \begin{bmatrix} \sum_{i=1}^m w_i\psi_i + Af \\ \sum_{i=1}^m w_i dx_i\psi_i + B_1 f + n_1\phi \\ \sum_{i=1}^m w_i dy_i\psi_i + B_2 f + n_2\phi \\ \sum_{i=1}^m w_i dz_i\psi_i + B_3 f + n_3\phi \\ \frac{1}{2}\sum_{i=1}^m w_i dx_i^2\psi_i + Cf \\ \sum_{i=1}^m w_i dx_i dy_i\psi_i \\ \sum_{i=1}^m w_i dx_i dz_i\psi_i \\ \frac{1}{2}\sum_{i=1}^m w_i dy_i^2\psi_i + Cf \\ \sum_{i=1}^m w_i dy_i dz_i\psi_i \\ \frac{1}{2}\sum_{i=1}^m w_i dz_i^2\psi_i + Cf \end{bmatrix}$$

Let $(\beta_0, \beta_1, ..., \beta_9)$ be the first row of matrix $(\tilde{M}^T \tilde{W} \tilde{M})^{-1}$. We are looking for functions $\psi = a_0$. Therefore, equating the first component of vectors on both sides of (7.13), we obtain

$$\begin{aligned}
\psi &= \beta_0 \left(\sum_{i=1}^{m} w_i \psi_i + Af \right) + \beta_1 \left(\sum_{i=1}^{m} w_i dx_i \psi_i + B_1 f + n_1 \phi \right) + \\
&\beta_2 \left(\sum_{i=1}^{m} w_i dy_i \psi_i + B_2 f + n_2 \phi \right) + \beta_3 \left(\sum_{i=1}^{m} w_i dz_i \psi_i + B_3 f + n_3 \phi \right) + \\
&\beta_4 \left(\frac{1}{2} \sum_{i=1}^{m} w_i dx_i^2 \psi_i + Cf \right) + \beta_5 \left(\sum_{i=1}^{m} w_i dx_i dy_i \psi_i \right) + \\
&\beta_6 \left(\sum_{i=1}^{m} w_i dx_i dz_i \psi_i \right) + \beta_7 \left(\frac{1}{2} \sum_{i=1}^{m} w_i dy_i^2 \psi_i + Cf \right) + \\
&\beta_8 \left(\sum_{i=1}^{m} w_i dy_i dz_i \psi_i \right) + \beta_9 \left(\frac{1}{2} \sum_{i=1}^{m} w_i dz_i^2 \psi_i + Cf \right)^T .
\end{aligned}$$

Rearranging the terms, we have

$$\begin{aligned}
\psi &= \sum_{i=1}^{m} w_i (\beta_0 + \beta_1 dx_i + \beta_2 dy_i + \beta_3 dz_i + \beta_4 \frac{dx_i^2}{2} + \beta_5 dx_i dy_i + \\
&\beta_6 dx_i dz_i + \beta_7 \frac{dy_i^2}{2} + \beta_8 dy_i dz_i + \beta_9 \frac{dz_i^2}{2}) \psi_i + \\
&(\beta_0 A + \beta_1 B_1 + \beta_2 B_2 + \beta_3 B_3 + \beta_4 C + \beta_7 C + \beta_9 C) f + (\beta_1 n_1 + \beta_2 n_2 + \beta_3 n_3) \phi.
\end{aligned}$$

Hence, if we consider $\mathbf{x}_j$ an arbitrary particle, $\mathbf{x}_{j_i}$ its neighbors of numbers $m(j)$, then we have the following sparse system of equations for the unknowns $\psi_j; j = 1, ...N$.

$$\begin{aligned}
\psi_j &= \sum_{i=1}^{m(j)} w_{j_i} (\beta_0 + \beta_1 dx_{j_i} + \beta_2 dy_{j_i} + \beta_3 dz_{j_i} + \beta_4 \frac{dx_{j_i}^2}{2} + \beta_5 dx_{j_i} dy_{j_i} + \\
&\beta_6 dx_{j_i} dz_{j_i} + \beta_7 \frac{dy_{j_i}^2}{2} + \beta_8 dy_{j_i} dz_{j_i} + \beta_9 \frac{dz_{j_i}^2}{2}) \psi_{j_i} + \\
&(\beta_0 A + \beta_1 B_1 + \beta_2 B_2 + \beta_3 B_3 + \beta_4 C + \beta_7 C + \beta 9C) f_j + \\
&(\beta_1 n_1 + \beta_2 n_2 + \beta_3 n_3) \phi_j. \qquad (7.14)
\end{aligned}$$

We can represent the above sparse system in compact matrix vector form as

$$\mathbf{A\Psi} = \mathbf{B} \qquad (7.15)$$

Hence, (7.15) is a big sparse linear system of equations and can be solved using iterative methods.

7.5 Weight Functions

In order to implement weighted least squares method to the elliptic PDE, we need the proper weight function $W = W(\mathbf{x}_i - \mathbf{x}; h)$ with small compact support h. The weight function can be quite arbitrary. The weight function is introduced in order to restrict the number of points around central particle where we want to approximate function and its derivatives. Different kinds of weight functions proposed recently by many researchers. It is observed that the accuracy, convergence and stability of some meshless methods depends on the weight functions in MLS approximation (see [PS03]). In [TK], authors have used a weight function of Gaussian type. The computations of flow problems using method of weighted least squares with Gaussian type weight function, can be found in [TK01], [TK02], [TJ02], [TK03], [Tiw00], [TM03]. In this section, we have made an attempt to give different weight functions and its effects on stability as well as accuracy of the system.

7.5.1 Weight Functions of Different Types

We have considered the weight functions of the following type:

1. Gaussian weight function of first kind

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} \exp\left(\frac{-\alpha d_i^2}{h^2}\right) & \text{if } d_i \leq h; \\ 0 & else. \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|,$$

α is a positive constant and h is the size of compact support.

This is the most commonly used weight function in literature of FPM method (see[TK01], [TK02], [TJ02], [TK03], [Tiw00], [TM03]) for computations of flow problems. They have used the value of α as 6.

2. Normal weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} \exp\left(\frac{-d_i^2}{2h^2}\right) & \text{if } d_i \leq h; \\ 0 & \text{else.} \end{cases}$$

where,

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

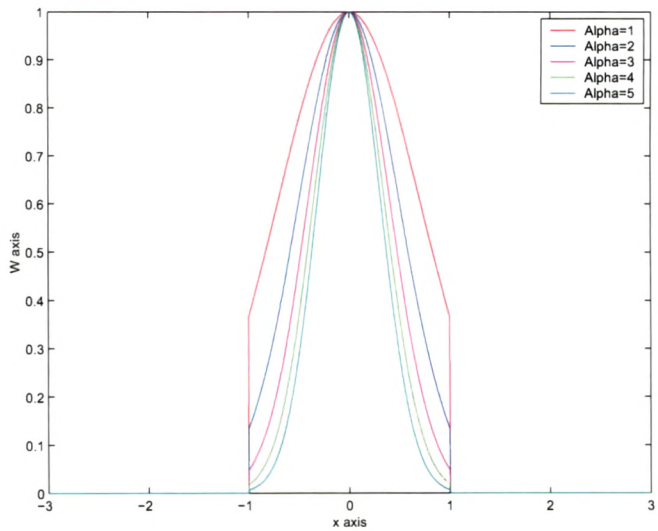

Figure 7.1: **Gaussian Weight Function of First Kind**

3. Triangular weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} 1 - \frac{|d_i|}{h} & \text{if } d_i \leq h; \\ 0 & \text{else.} \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

4. Quadratic weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} 1 - \frac{(d_i)^2}{h^2} & \text{if } d_i \leq h; \\ 0 & \text{else.} \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

5. Tri-cubic weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} \left(1 - \frac{|d_i|^3}{h^3}\right)^3; & \text{if } d_i \leq h \\ 0; & \text{else.} \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

6. Epanecanikov weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} 0.75\left(1 - \frac{(d_i)^2}{h^2}\right) & \text{if } d_i \leq 1; \\ 0 & \text{else.} \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

7. Zhou W.Y. and Kou X.D weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} \frac{h^2}{d_i^2 + h^2\epsilon^2}\left(1 - \frac{d_i^2}{h^2}\right)^k & \text{if } d_i \leq h; \\ 0 & \text{if } d_i \geq h. \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|, \epsilon = 0.5$$

and $k = 4$ and r is the radius of support.

8. Gaussian weight function of second kind

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} \frac{\exp\left(-\left(\frac{d_i}{c}\right)^{2k}\right) - \exp\left(-\left(\frac{h}{c}\right)^{2k}\right)}{1 - \exp\left(-\left(\frac{h}{c}\right)^{2k}\right)} & \text{if } d_i \leq h; \\ 0 & \text{if } d_i \geq h. \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|,$$

c is the parameter controlling the shape of weight function. We shall take $c = 1$ and $k = 1$.

9. Exponential weigh weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} \exp\left(-\left(\frac{d_i}{\alpha h}\right)\right) & \text{if } d_i \leq h; \\ 0 & \text{if } d_i \geq h. \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|,$$

α is a parameter. We shall take its value is 0.5.

10. Cubic spline weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} \frac{2}{3} - 4\left(\frac{d_i}{h}\right)^2 + 4\left(\frac{d_i}{h}\right)^3 & \text{if } d_i \leq h/2; \\ \frac{4}{3} - 4\frac{d_i}{h} + 4\left(\frac{d_i}{h}\right)^2 - \frac{4}{3}\left(\frac{d_i}{h}\right)^3 & \text{if } h/2 \leq d_i \leq h; \\ 0 & \text{if } d_i \geq h. \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \| .$$

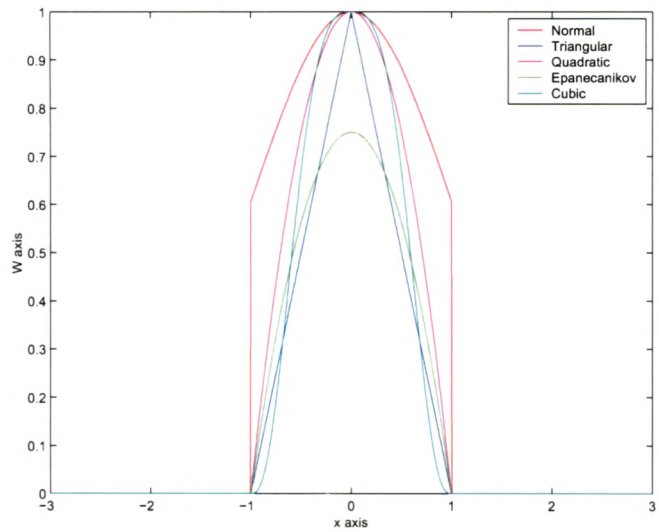

Figure 7.2: **Different weight functions- I**

11. Quartic spline 1-weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \begin{cases} 1 - 6\left(\frac{d_i}{h}\right)^2 + 8\left(\frac{d_i}{h}\right)^3 - 3\left(\frac{d_i}{h}\right)^4 & \text{if } d_i \leq h; \\ 0 & \text{if } d_i \geq h. \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

12. Ours weight function

$$W(\mathbf{x}_i - \mathbf{x}; r) = \begin{cases} \left(1 - (d_i)^2\right)^{\alpha} & \text{if } 0 \leq d_i \leq h; \\ 0 & \text{else.} \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

and α is any constant.

13. Quartic spline 2-weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \frac{8}{115} \begin{cases} \left(\frac{5}{2}\right)^4 \left(1 + \frac{d_i}{h}\right)^4 - 5\left(\frac{1}{2}\right)^4 \left(3 + 5\frac{d_i}{h}\right)^4 + 10\left(\frac{1}{2}\right)^4 \left(1 + 5\frac{d_i}{h}\right)^4 & \text{if } 0 \leq d_i \leq h/5; \\ \left(\frac{5}{2}\right)^4 \left(1 - \frac{d_i}{h}\right)^4 - 5\left(\frac{1}{2}\right)^4 \left(3 - 5\frac{d_i}{h}\right)^4 & \text{if } h/5 \leq d_i \leq 3h/5; \\ \left(\frac{5}{2}\right)^4 \left(1 - \frac{d_i}{h}\right)^4 & \text{if } 3h/5 \leq d_i \leq h; \\ 0 & \text{if } h \leq d_i. \end{cases}$$

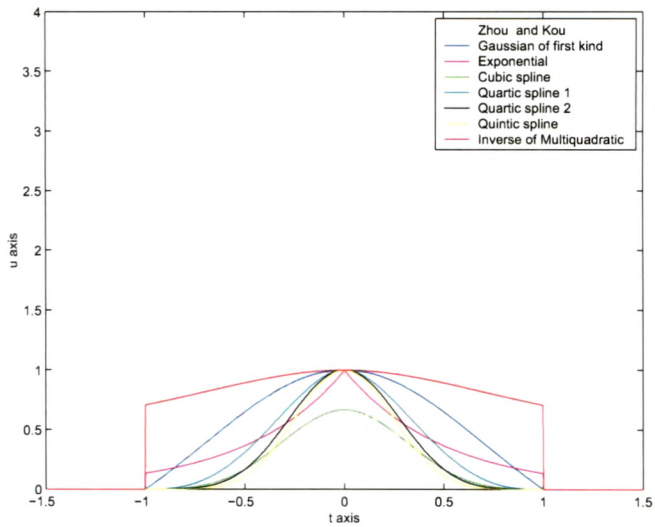

Figure 7.3: **Different weight functions- II**

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

14. Quantic spline weight function

$$W(\mathbf{x}_i - \mathbf{x}; h) = \frac{1}{66} \begin{cases} \left(3 - 3\frac{d_i}{h}\right)^5 - 6\left(2 - 3\frac{d_i}{h}\right)^5 + 15\left(1 - 3\frac{d_i}{h}\right)^5 & \text{if } 0 \leq d_i \leq h/3; \\ \left(3 - 3\frac{d_i}{h}\right)^5 - 6\left(2 - 3\frac{d_i}{h}\right)^5 & \text{if } h/3 \leq d_i \leq 2h/3; \\ \left(3 - 3\frac{d_i}{h}\right)^5 & \text{if } 2/3 \leq d_i \leq h; \\ 0 & \text{if } h \leq d_i. \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|$$

15. Inverse of Multiquadratic weight function

$$W(\mathbf{x}_i - \mathbf{x}; r) = \begin{cases} \frac{1}{\sqrt{1+\alpha\left(\frac{d_i}{h}\right)^{2k}}} & \text{if } 0 \leq d_i \leq h; \\ 0 & \text{if } h \leq d_i. \end{cases}$$

where

$$d_i = \| \mathbf{x} - \mathbf{x}_i \|, \quad \alpha = 1, \quad k = 1$$

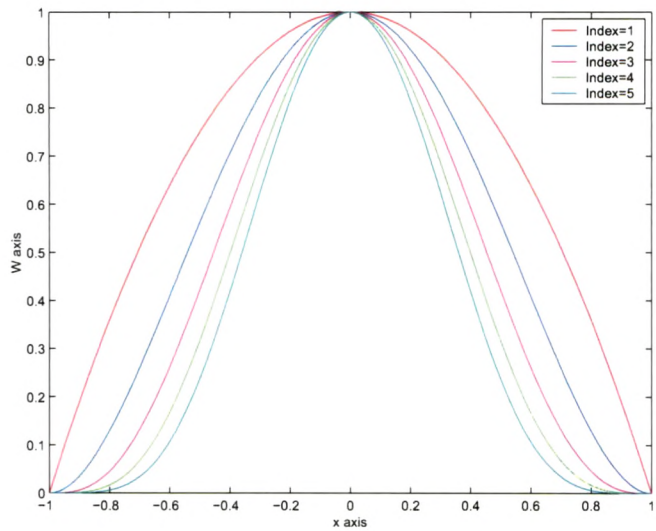

Figure 7.4: **Ours weight function**

7.5.2 Influence of Weight Function on Error and Condition Number

In this section, we shall analyze the impact of different weight functions on error and condition number of final big sparse matrix resulting from FPM discretization, given in equation (7.15). The convergence study of this FPM scheme shows that it is of second order convergence using Gaussian weight function. However, it is observed that sometimes it is difficult to invert a small matrix (M^TWM) as well as big matrix A in order to get a solution of a system. In this section, we have made an attempt to do the following analysis using test examples in $1D$ as well as $2D$:

- effect of different weight functions on condition number of a matrix.
- error analysis using different weight functions.

Finally, we shall show that which weight function gives better conditioned system as well as good accuracy.

7.6 Numerical Results

7.6.1 Test Examples in $1D$

Dirichlet problem

Consider the following Dirichlet problem

$$\frac{d^2y}{dx^2} + 4\frac{dy}{dx} + 4y = \exp x; \quad y(0) = 1, \quad y(1) = 0. \tag{7.16}$$

Neumann problem

Consider the following Neumann problem

$$\frac{d^2y}{dx^2} + 4\frac{dy}{dx} + 4y = x^2 + 1; \quad \frac{dy}{dx}(0) = 0, \quad \frac{dy}{dx}(1) = 0 \tag{7.17}$$

Mixed boundary value problem

Consider the following mixed problem

$$\frac{d^2y}{dx^2} + 4\frac{dy}{dx} + 4y = \exp x; \quad y(0) = 1, \frac{dy}{dx}(1) = 0 \tag{7.18}$$

7.6.2 Test Examples in $2D$

Dirichlet problem

Consider the following Dirichlet problem. Let Ω be a domain.

$$\Omega = \{(x, y) | -1 < x < 1; \quad and \quad -1 < y < 1\}$$

$$u_{xx} + u_{yy} = 4; \quad on \ \ \Omega \tag{7.19}$$

$$u = x^2 + y^2; \quad on \quad \partial\Omega \tag{7.20}$$

Neumann problem

Consider the following Neumann problem. Let Ω be a domain.

$$\Omega = \{(x, y) | 0 < x < 1, \ and \ 0 < y < 1\}$$

$$u_{xx} + u_{yy} = -\cos(\pi x), \ on \ \Omega \tag{7.21}$$

$$\frac{\partial u}{\partial \mathbf{n}} = 0, \ on \ \partial\Omega \tag{7.22}$$

Helmholtz boundary value problem

Consider the following mixed problem. Let Ω be a domain.

$$\Omega = \{(x, y) | -1 < x < 1, \ and \ -1 < y < 1\}$$

$$u_{xx} + u_{yy} + u = 4 + x^2 + y^2, \ on \ \Omega \tag{7.23}$$

$$\frac{\partial u}{\partial \mathbf{n}} = 2y, \ on \ y = -1,$$

$$\frac{\partial u}{\partial \mathbf{n}} = -2y, \ on \ y = 1,$$

$$\frac{\partial u}{\partial \mathbf{n}} = 2x, \ on \ x = -1,$$

$$\frac{\partial u}{\partial \mathbf{n}} = -2x, \ on \ x = 1.$$

7.6.3 Results and Discussion

Conclusion for conditioned system in both dimension

- Gaussian function of first kind gives very high condition number in compare to all other weight functions. Since condition number is very high for this weight function, we have not presented it in the figure, otherwise analysis of other weight functions can not be analyzed in a proper way.
- Quantic spline, Quartic Spline 2, Gaussian of second kind, Ours, Normal, Epanecanikov, Inverse Multiquadratic, Quadratic, Tricubic, Exponential, Triangular and Quartic Spline 1 are in hierarchy for better condition number.

Conclusion for accuracy of system

- As support of weight function increase error increases, as expected, but Gaussian weight function gives better accuracy. Only disadvantage of this weight function is the high condition number.
- Tricubic is oscillating.
- From error point of view hierarchy is cubic spline, Triangular, Zhou and Kou, Gaussian, Inverse Multiquadratic, Quadratic, Quartic spline, Ours, Exponential, Normal, Quartic spline 2, and Quantic spline.