HIV-Protease Dynamics using Scattered Data Approximation

The University of New Mexico

Department of Mathematics and Statistics

Undergraduate Honors Thesis

Student: Rodrigo Osuna Orozco

Adviser: Prof. Deborah Sulsky

Abstract

HIV protease is fundamental in the replication of the human immunodeficiency virus in the organism and has thus been a target for the development of many drugs. HIV-PR inhibitors' effectiveness is hampered by the emergence of resistant mutants. Thus understanding HIV-PR dynamics has become of utmost importance. We use the results from a molecular dynamics simulation of HIV-PR. The simulation gives the atoms' positions at discrete, regular time intervals. We then perform scattered data approximations to examine displacements and deformations at uniformly spaced points on a regular grid. Our approximations show that regions of high deformation can be recognized at spatial scales comprising about ten inter-atomic distances. Thus it is possible to characterize some salient features of the HIV protease's dynamics at scales in which the motion of individual atoms is not considered in full detail. This could offer some computational advantage as well as allowing to understand the role of the protease's flexibility in its dynamics, and ultimately in functionality.

Table of Contents

1) Introduction

1.1) Background on HIV-PR

The aspartic protease of human immunodeficiency virus (HIV-PR) is an enzyme critical for the replication of HIV virions. The function this enzyme performs is indispensable for the production of infectious viral particles; hence, if inhibited, the spread of HIV through the organism is halted (Pokorná et al, 2009). This situation has made HIV-PR an especially attractive target for the development of anti-HIV drugs, and the introduction of such drugs has helped foster a significant decrease in the mortality rates of HIV infected patients (figure 1) (Pokorná et al, 2009).

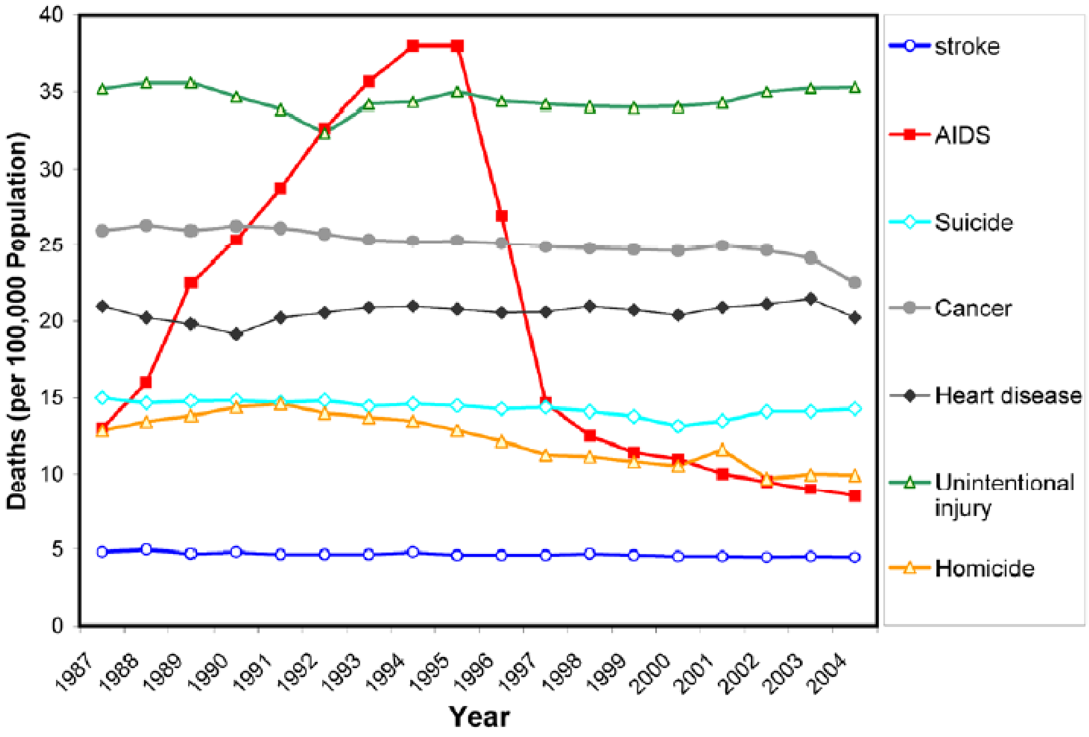


Figure 1: From Pokorna et al. Trends in annual rates of death due to 7 leading causes among persons 25-44 years old in the United States during period 1987-2004. Dramatic decrease in the rate of deaths due to AIDS coincides with the introduction of HIV protease inhibitors.

Nonetheless, in spite of this initial success, the effectiveness of HIV-PR inhibitors is limited by the emergence of drug-resistant and cross resistant mutants (Brik and Wong 2003). Thus an understanding of the mechanisms through which HIV-PR performs its functions has become essential in order to create drugs that may be effective against a host of possible mutations. In particular, it has been noted that a greater understanding of the dynamics associated to the binding of inhibitors and substrates to HIV-PR is crucial (Hornak and Simmerling, 2007). Indeed, the enzyme undergoes significant conformational changes when binding to a substrate and its dynamics offer an explanation as to how inhibitors reach HIV-PR's binding site (see figure 2) (Hornak and Simmerling, 2007). Thus it has been pointed out that the enzyme's flexibility may play an important role in generating resistance to inhibitors (variations in flexibility can enhance the activity of the enzyme) (Hornak and Simmerling, 2007).

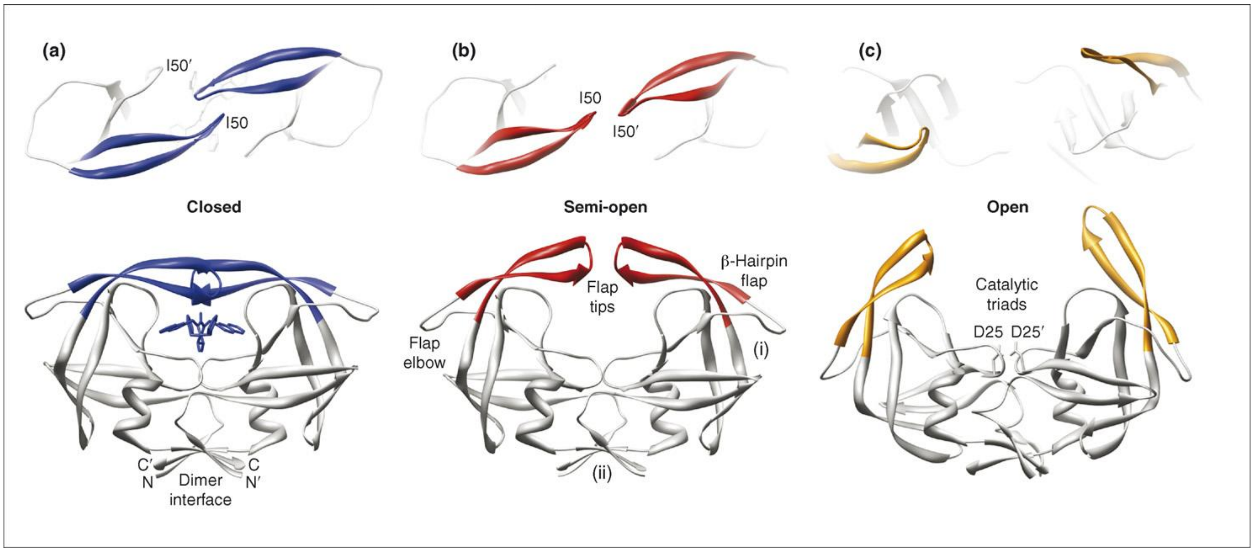


Figure 2: From Hornak and Simmerling. Images from molecular dynamics simulations showing three important conformations of HIV-PR. Inhibitors may reach the active site when the enzyme is in the fully open configuration.

However, the roles of dynamics and flexibility have not received much attention on behalf of the modeling community (Wong and McCammon, 2003; Hornak and Simmerling, 2007). And, an additional complication arises as modeling and predicting the kinematics of large domains is still a significant challenge in computational drug design (Carlson and McCammon, 2000).

1.2) Motivation for our approach

As indicated in the preceding section the mechanical behavior of HIV-PR determines its functional properties to a large extent. Nonetheless, as even the simplest of microorganisms (viruses) are comprised by several thousands of atoms which arrange themselves as complex molecules; a detailed description which traces the kinematics of each single atom quickly grows in complexity. Such complexity not only has a significant computational cost associated to it but it may also obscure the overall behavior of the system. Thus in order to gain some insight, we may consider characterizing an enzyme's average motions over length scales larger than the inter-atomic distances. Furthermore we may seek to study the deformation of the molecule by looking at quantities commonly used in continuum mechanics to represent strains, deformation gradients and stresses.

Such an approach may facilitate the study of large deformations (and conformational changes) in HIV-PR as well as allowing the development of insight regarding the role of flexibility in the molecule's dynamic behavior. This approach can help establish a link between changes in flexibility, deformation, dynamics and ultimately drug resistance.

1.3) Mathematical Statement of the Problem

Considering the situation described above we formulate the following problem. Given N_p (non uniform) data points with coordinate vectors $\boldsymbol{x}_p(t), p = 1, 2, 3, \ldots, N_p$ and data values $\boldsymbol{u}_p(t)$ at those points, we view $\boldsymbol{u}_p(t)$ as sampled values of an underlying function $\boldsymbol{u}(\boldsymbol{x}, t)$. That is, we would like to construct a function $\boldsymbol{u}(\boldsymbol{x}, t)$, so that $\boldsymbol{u}(\boldsymbol{x_p}, t)$ is approximately $\boldsymbol{u}_p(t)$. Since we expect the motion of atoms in HIV-PR to have a mean component and fluctuations about a mean, we want to regard $\boldsymbol{u}(\boldsymbol{x}, t)$ as representing the mean. To extract the mean motion associated with a given length scale, we will examine $\boldsymbol{u}(\boldsymbol{x}, t)$ at a set of N_g uniformly distributed grid points.

. Thus we look for a function that can be evaluated at any point in a domain which contains all data sites x_p, and that is at least once differentiable (this serves the purpose of evaluating gradients in displacement). Moreover we would like to ensure that the smoothness of our resulting function is greater than that of the underlying function that generated the data originally.

2) A first approximation

A first approximation to the problem stated above can be obtained by considering averaging a compact subset of the data values around a given grid point. We can reasonably propose a weighted average around a grid point where the weight function has a compact support, and decreases with increasing distance from the grid point.

Before attacking the problem of HIV-PR, let's consider sampling a known function at a random set of data sites and trying to reconstruct the known function values at grid points from these scattered data.

We can state the one dimensional problem as follows. Let $f: [0, L] \to \mathbb{R}$ be the function whose values are known at a set of data points $x_p \in [0, L]$, we approximate the values of f at regularly spaced points $x_g = j\Delta x, j = 0,1,2,3, \dots N_g, \Delta x = \frac{L}{N_g}$, where $N_g \in \mathbb{N}$ is the number of grid elements into which we divide the set $[0, L]$. The approximated values at the grid points:

(1) $$\tilde{f}(x_g) = \frac{\sum_{p=1}^{Np} f(x_p)\omega(x_p - x_g)}{\sum_{p=1}^{Np} \omega(x_p - x_g)}$$

Where

(2) $$\omega(x - x_g) = \begin{cases} 1 - \frac{|x - x_g|}{\Delta x}, & \frac{|x - x_g|}{\Delta x} \leq 1 \\ 0, & otherwise \end{cases}$$

Figure 3 illustrates the weight function ω used in the examples below. Data from the two neighboring elements that surround a grid point are used to obtain $\tilde{f}(x_g)$.

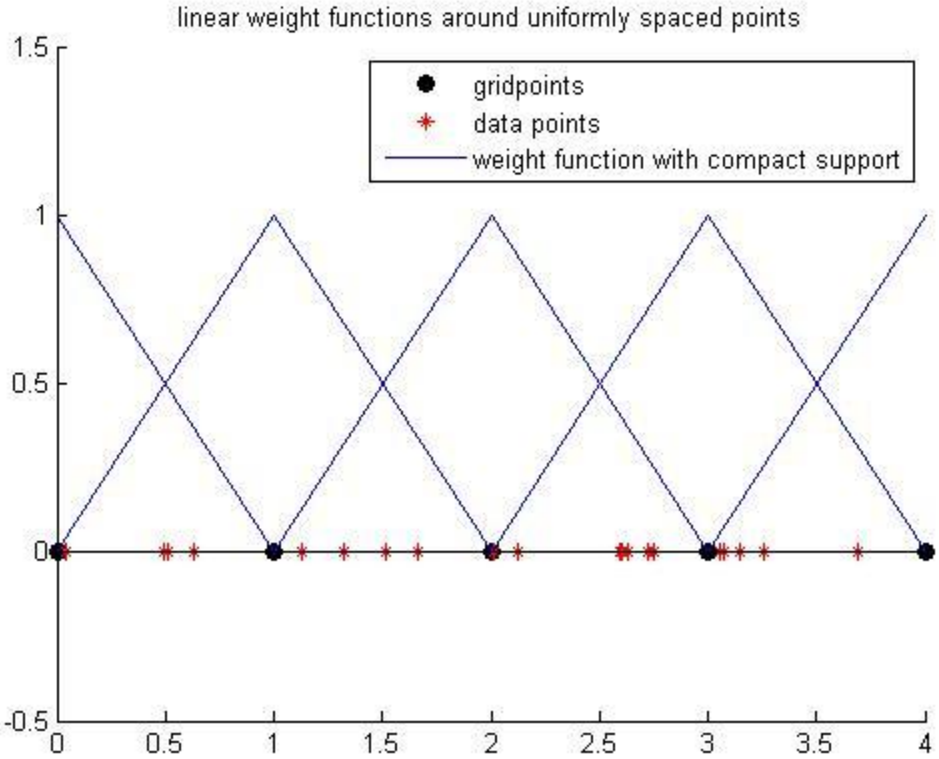


Figure 3: Uniformly distributed 1-D grid points with linear weight functions with constant support. The support of the weight functions is given as twice the distance between the grid points.

A clear property of such approximation, called Shepard’s function due to its proposer (Shepard, 1968), is that it reproduces constants. That is, if $f(x) = c$, then $\tilde{f}(x) = c$ also, where c is a constant. In fact for a given number of uniformly distributed random data points we see that the error of our approximation increases as the underlying function that generated the data deviates from constant (figure 4). In principle we could approximate a wide variety of functions just by considering a fine enough grid. However, this is not feasible in practice as we consider the case of having a finite set of data points.

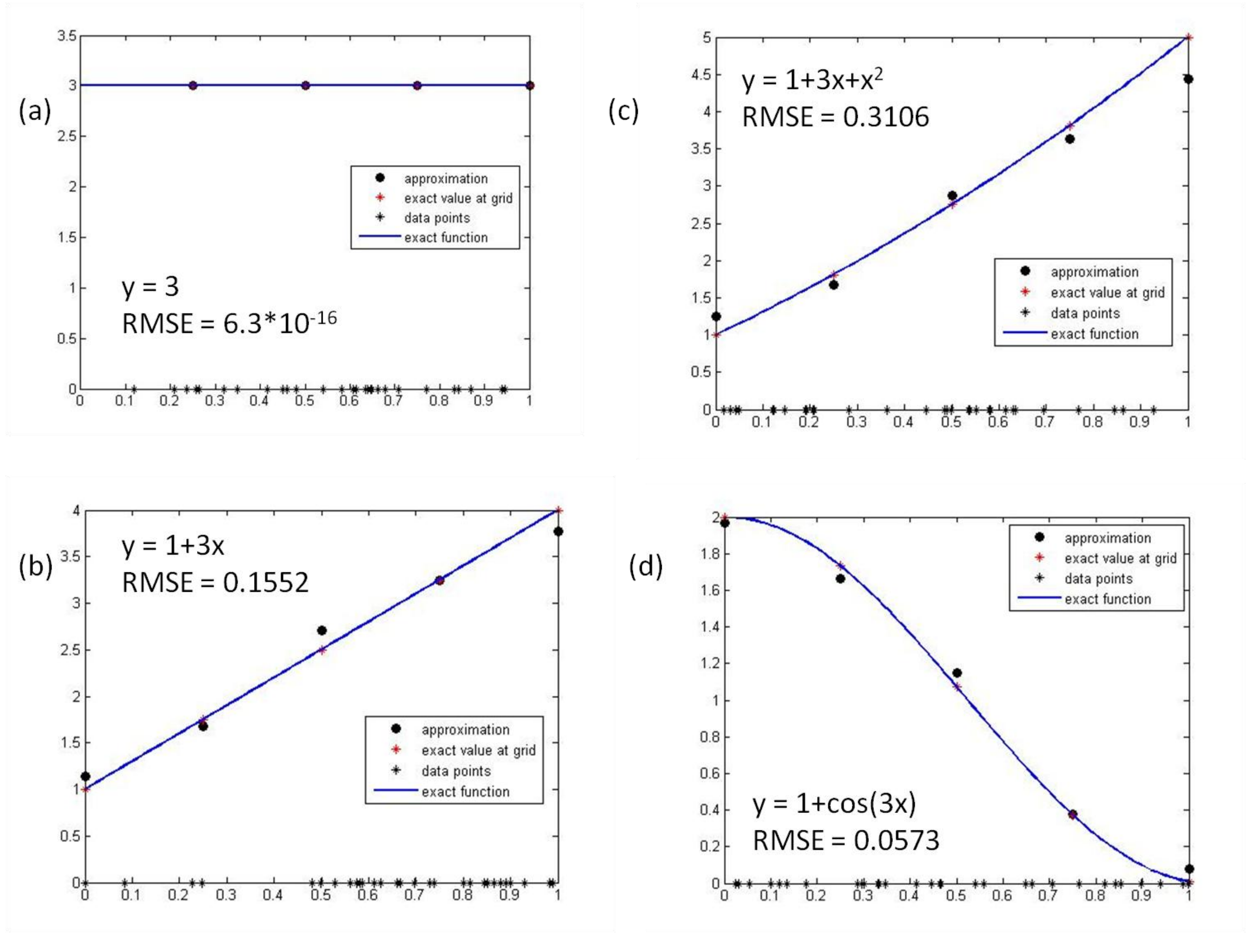


Figure 4: Shepard's function applied to one-dimensional examples. Note that the root mean squared errors increase as the approximated functions deviate from constant (in particular as their image spans a larger set). These examples were done using values N_g=4 and N_p = 30.

In the examples in figure 4 we have used as weight functions the functions with compact support illustrated in figure 3.

This idea can easily be generalized for arbitrary dimensions. Clearly there is no need to modify equation 1 above, so we are only left with the choice of weight function to consider. For this purpose we multiply linear weight functions of the form of equation

2 for each of the orthogonal directions. Figure 5 illustrates the weight function in two dimensions. Thus we can express the weight function in higher dimensions as the product of weight function for its components, namely:

(3) $$\omega\left(\boldsymbol{x}-\boldsymbol{x}_g\right)=\omega\left(x-x_g\right)\omega\left(y-y_g\right) \text{ in 2D}$$

(4) $$\omega\left(\boldsymbol{x}-\boldsymbol{x}_g\right)=\omega\left(x-x_g\right)\omega\left(y-y_g\right)\omega\left(z-z_g\right) \text{ in 3D}$$

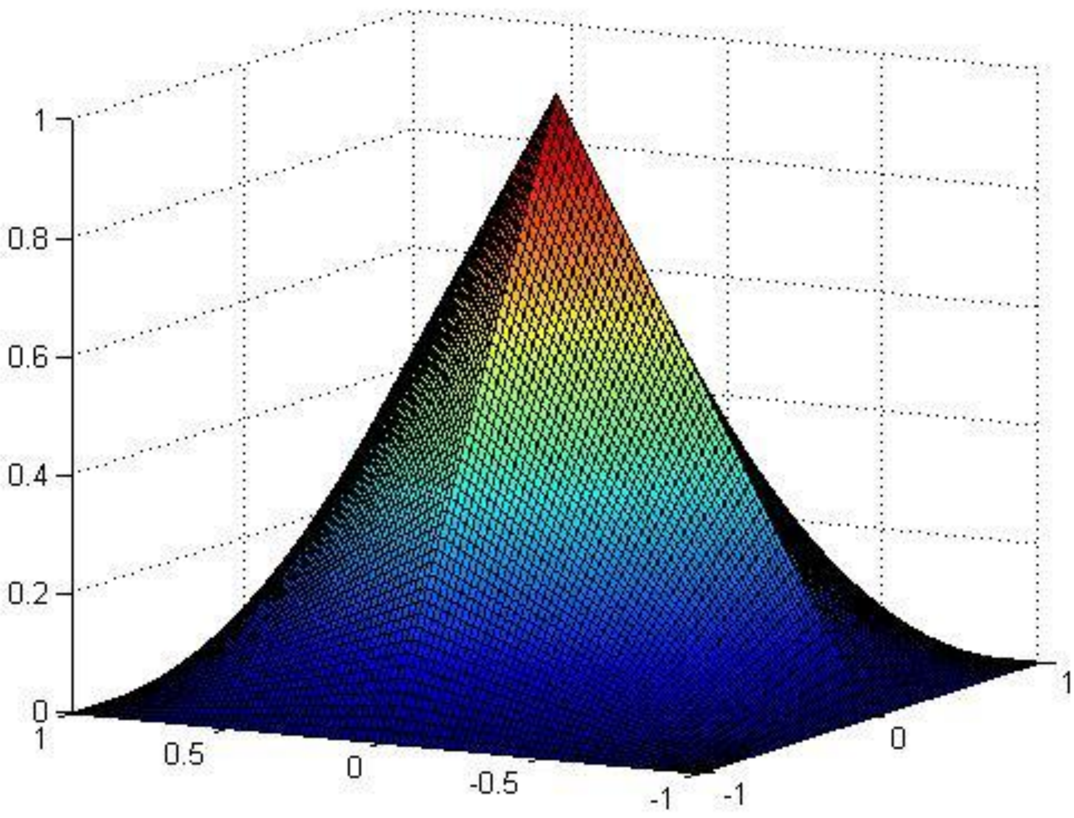


Figure 5: weight function in 2D (equation 3) with compact support equal to a unit square around the origin. We see how the weight decreases away from the center and has 1-norm radial symmetry.

We may notice that the weight functions being considered here are radial weight functions. Namely they satisfy the following definition:

A function is $\Phi: \mathbb{R}^d \to \mathbb{R}$ is called radial provided there exists a univariate function $\phi: [0, \infty) \to \mathbb{R}$ such that

$$\Phi(\boldsymbol{x}) = \phi(r) \text{ , where } r = ||\boldsymbol{x}||$$

And $||\cdot||$ is some norm on $\mathbb{R}^d$. (Fasshauer, 2007).

In particular our weight functions are radial under the one norm (figure 6).

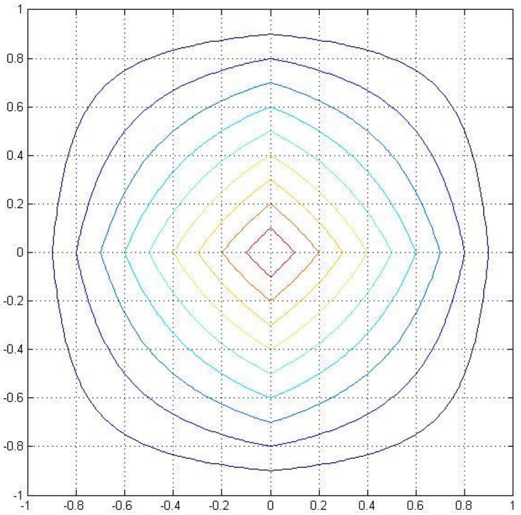


Figure 6: Iso-contours for equation 3 centered at the origin. Notice that the contours denote regions where the sum of the x and y coordinates is constant.

Figure 7 illustrates the use of Shepard's function approximation in two dimensions. As before we can observe how our approximation reproduces a nonlinear function fairly well if the distribution of data points is dense enough.

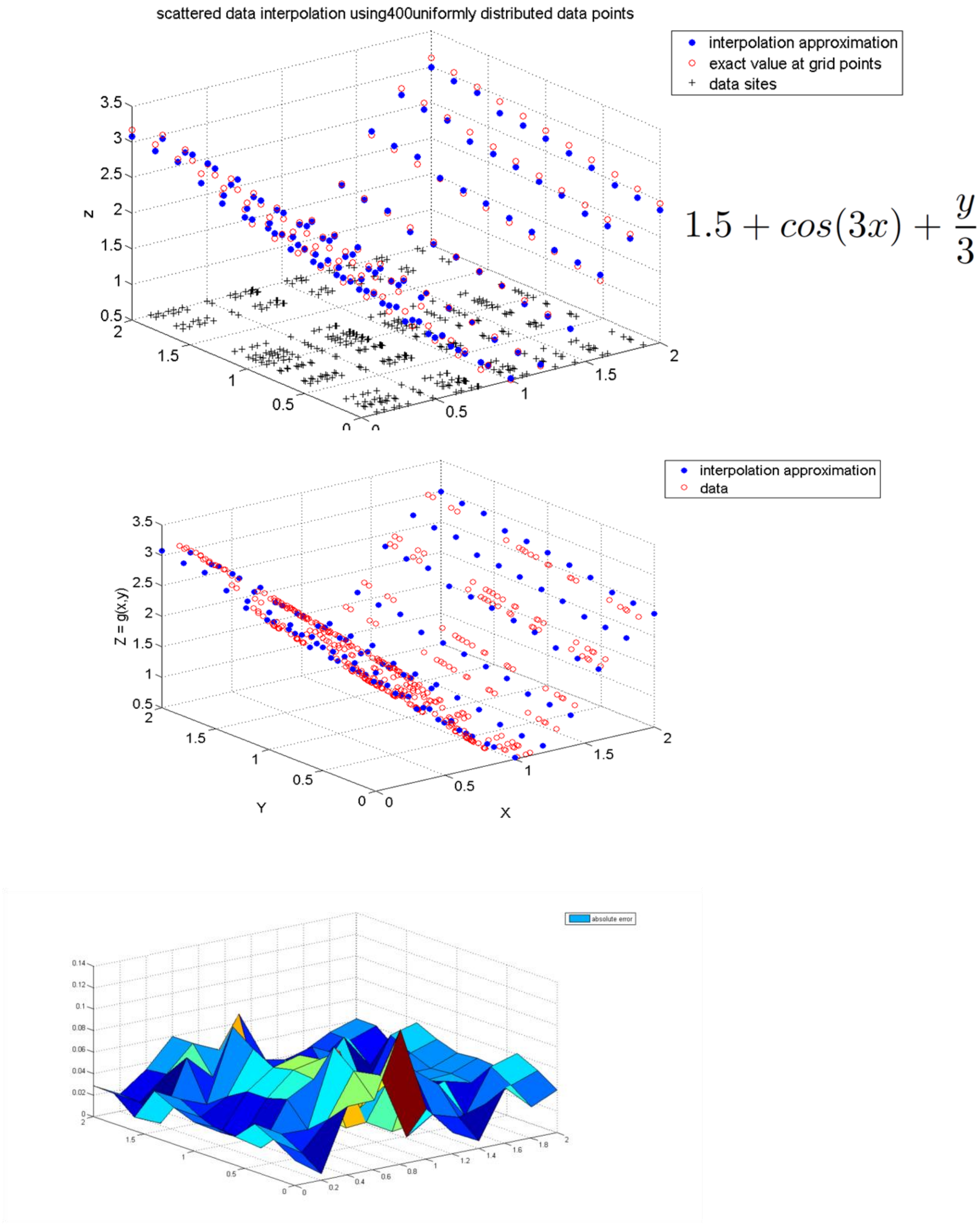


Figure 7: 2D example of approximation using Shepard's function. At the top we see the approximated values, the actual values of the function at the grid points and the distribution of data points. In the middle we see the original data and the approximations. In the bottom we see a plot of the relative error which is about 10%. These examples were done using values N_p = 400 and N_g =N_{gx}*N_{gy} =10*10 (where N_{gx} and N_{gy} are the number off grid cells in the x and y directions).

2.1) Application to Molecular Dynamics Simulation Data

We now want to apply Shepard's method to scattered data for the HIV-PR. As our scattered data we use the results of molecular dynamics simulations courtesy of Carlos Simmerling's Lab. Here our data sites are given by the atoms' positions and the data at those sites by the velocities and displacements at those atoms. We are given trajectories of the atoms as a set of positions at regular time intervals of around 1 microsecond. If the vector of the p^th atom's coordinates at time t is given by $\boldsymbol{x}_p(t)$ and the time interval is given by Δt we can estimate the velocities and the displacements as follows:

$$\boldsymbol{u}_p\left(\boldsymbol{x}_p, t\right) = \frac{\boldsymbol{x}_p(t) - \boldsymbol{x}_p(t-\Delta t)}{\Delta t}$$ is the velocity of the pth atom at time t.

$$\boldsymbol{U}_p\left(\boldsymbol{x}_p, t\right) = \boldsymbol{x}_p(t) - \boldsymbol{x}_p(t=0)$$ is the displacement of the pth atom at time t.

Thus we get a set of scattered data points at regular time steps from which we want to approximate displacement and velocity fields which will be evaluated on our uniform grid (figure 8). We use Matlab to implement Shepard's method for our data set and visualize the results in paraview (See Appendix A and C).

Thus we think of $\boldsymbol{u}_p$ and $\boldsymbol{U}_p$ as sampled values of velocity and displacement functions respectively, $\boldsymbol{u}, \boldsymbol{U}: \Omega \to \mathbb{R}^3$, where $\Omega \subset \mathbb{R}^3$ is a domain containing the atoms for all times of interest. We enclose the atoms in a box and thus, $\Omega = [-33.5, 36] \times [-27.2, 30.4] \times [-21.6, 22.6]$ and $\in [0, T]$. And we apply Shepard's function (equation 1) with equation 4 as our weight function. Figure 9 shows an example of this procedure for approximating displacement vectors on a grid consisting 6x6x6 elements on the ninth time interval; the top images show the atoms' and grid locations, and the bottom images show the displacement vectors colored by their magnitude at these locations.

With such approximations in hand it is possible to use finite differences to estimate the value of the displacement gradient at the centers of our grid elements. For each grid cell, we introduce a local numbering of the grid points as indicated in figure 8. The cell center is $\boldsymbol{x}_c = \frac{1}{8}\sum_{g=1}^{8} \boldsymbol{x}_g$.

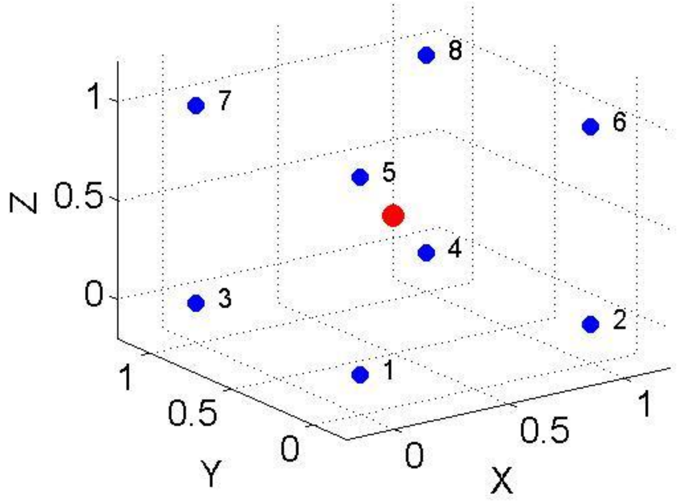


Figure 8: Grid element with corners in blue and center in red. Derivatives are calculated at the center using values at the corners as a centered difference.

We can approximate the value of derivatives at the center of a grid element by:

(5) $$\frac{\partial u_c}{\partial x} = \frac{1}{4}\left(\frac{u_2 - u_1}{\Delta x} + \frac{u_4 - u_3}{\Delta x} + \frac{u_6 - u_5}{\Delta x} + \frac{u_8 - u_7}{\Delta x}\right)$$

(6) $$\frac{\partial u_c}{\partial y} = \frac{1}{4}\left(\frac{u_3 - u_1}{\Delta x} + \frac{u_4 - u_2}{\Delta x} + \frac{u_7 - u_5}{\Delta x} + \frac{u_8 - u_6}{\Delta x}\right)$$

$$(7) \qquad \frac{\partial u_c}{\partial z} = \frac{1}{4}\left(\frac{u_5 - u_1}{\Delta x} + \frac{u_6 - u_2}{\Delta x} + \frac{u_7 - u_3}{\Delta x} + \frac{u_8 - u_4}{\Delta x}\right).$$

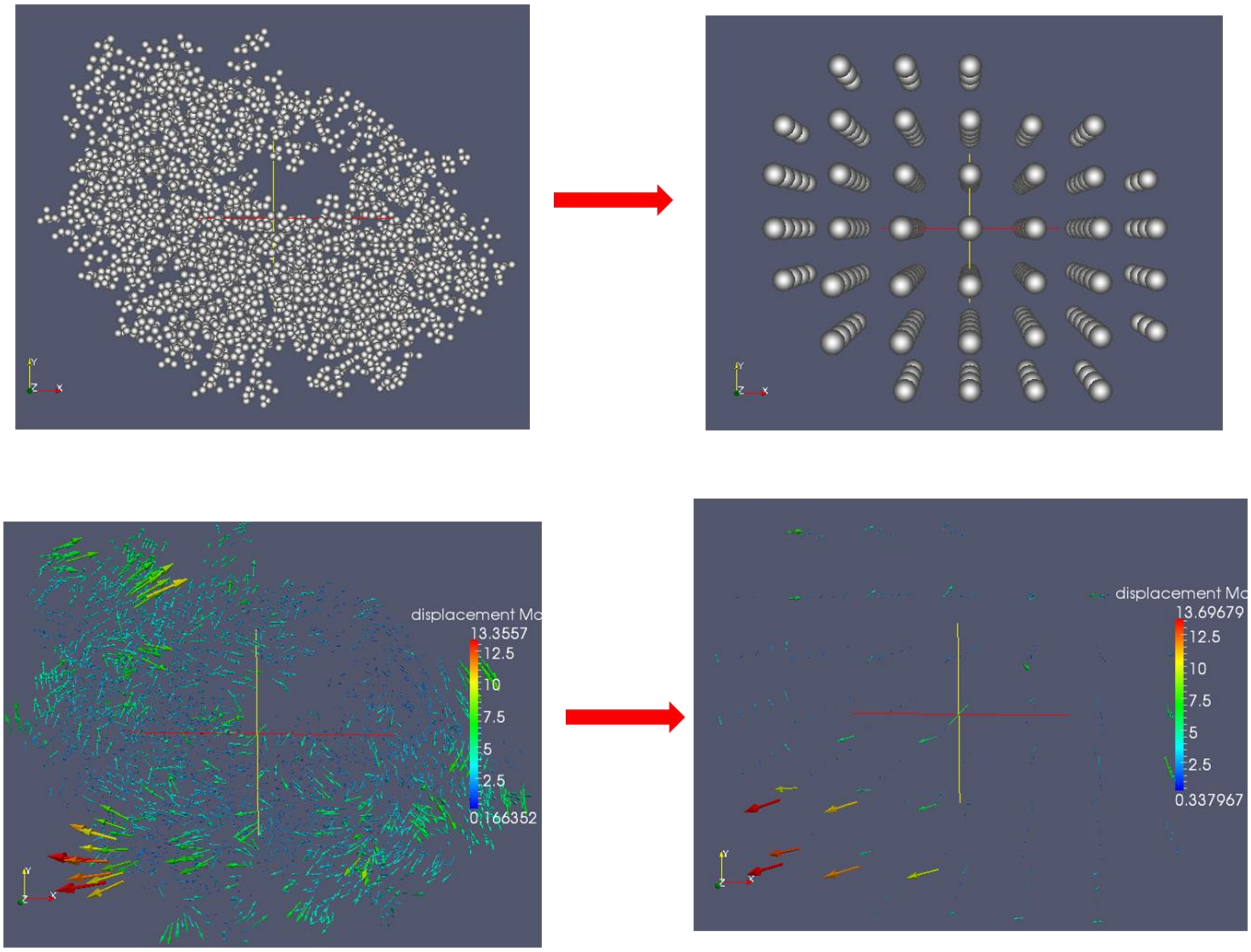


Figure 9: Top: atoms' positions (left) and grid points positions (right); notice how much sparser our grid is compared to the original distribution of atoms. Bottom: displacement at atoms' locations (left) and at grid points (right), the displacement vectors are colored by displacement magnitude.

From these finite difference approximations of derivatives, we can construct various strain measures. In continuum mechanics $\boldsymbol{U}(\boldsymbol{X}, t)$ is the displacement at time t of a particle that started at position **X** at time t=0. Measures of deformation are given by (see Appendix B on strain):

Displacement gradient --------------------------------- $\nabla_0 U$

Deformation gradient tensor ------------------------- $F = I + \nabla_0 U$

Green strain Tensor ------------------------------------ $G = \frac{1}{2}(F^T F - \mathrm{I})$

Linear strain tensor --------------------------------------- $E = \frac{1}{2}(\nabla_0 U + \nabla_0 U^{\mathrm{T}})$

Where the derivatives are taken with respect to the components of **X**. These quantities are second order tensors.

Clearly we only want to consider deformations and not displacements related to solid body translations or rotations. Moreover, initially we only consider the “magnitude” of those deformations; hence we look for a suitable norm. The Frobenius norm for a matrix is defined as the square root of the sum of the squares of the values of its elements, namely:

$$||A||_{Fro} = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2}$$

Figure 10 shows an example where grid point values are colored by the Frobenius norm of the linear strain, E. Higher values of linear strain are associated with regions of the enzyme having larger displacements (also shown in figure 10). We observe that even for length scales that are about ten times the inter-atomic distances we get regions of high deformation (particularly noticeable near the flaps and edges) corresponding to regions of high strain on our regular grid.

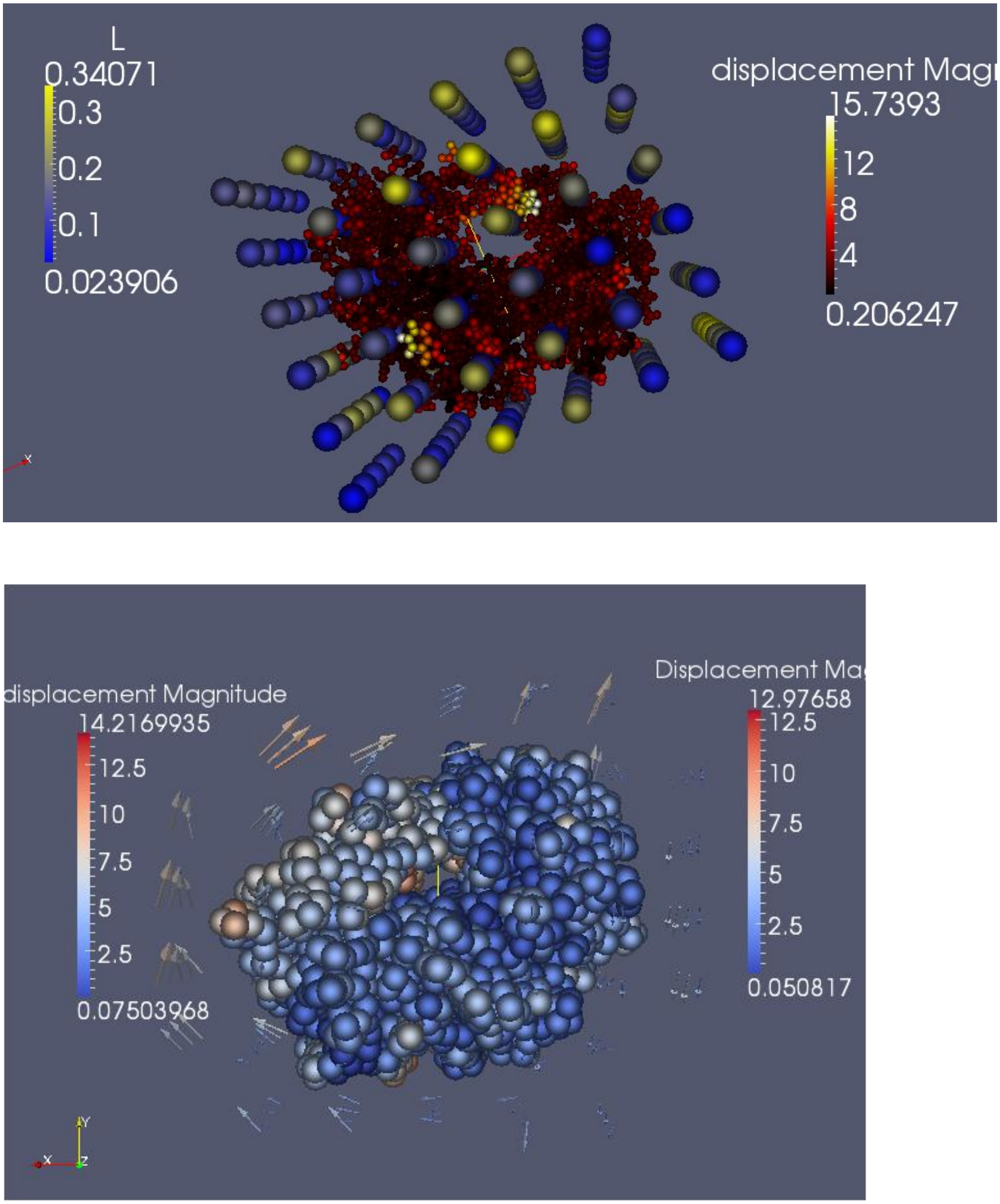


Figure 10: (Top) centers of grid elements are represented by the large spheres and are colored by the Frobenius norm of linear strain. The small spheres represent the atoms' locations and colored by their displacement magnitude. Regions of high deformation on the flaps correspond to regions of high linear strain. (Bottom) Regions of high displacement in the molecule (atoms are shown as spheres) appear also as regions of high displacement in our sparse grid (displacement vectors colored by magnitude are shown for grid points). The results shown are for a grid consisting of 6x6x6 elements at the 50th time interval.

Moreover for large regions within the enzyme the difference between the linear and Green strains is not too large at a variety of length scales (figure 11) which indicates that the deformation for the HIV-PR can be described by the linear strain for most of the HIV-PR. However Regions of high deformation become more noticeable as we consider scales closer to the inter-atomic distances, hence the relative difference (computed

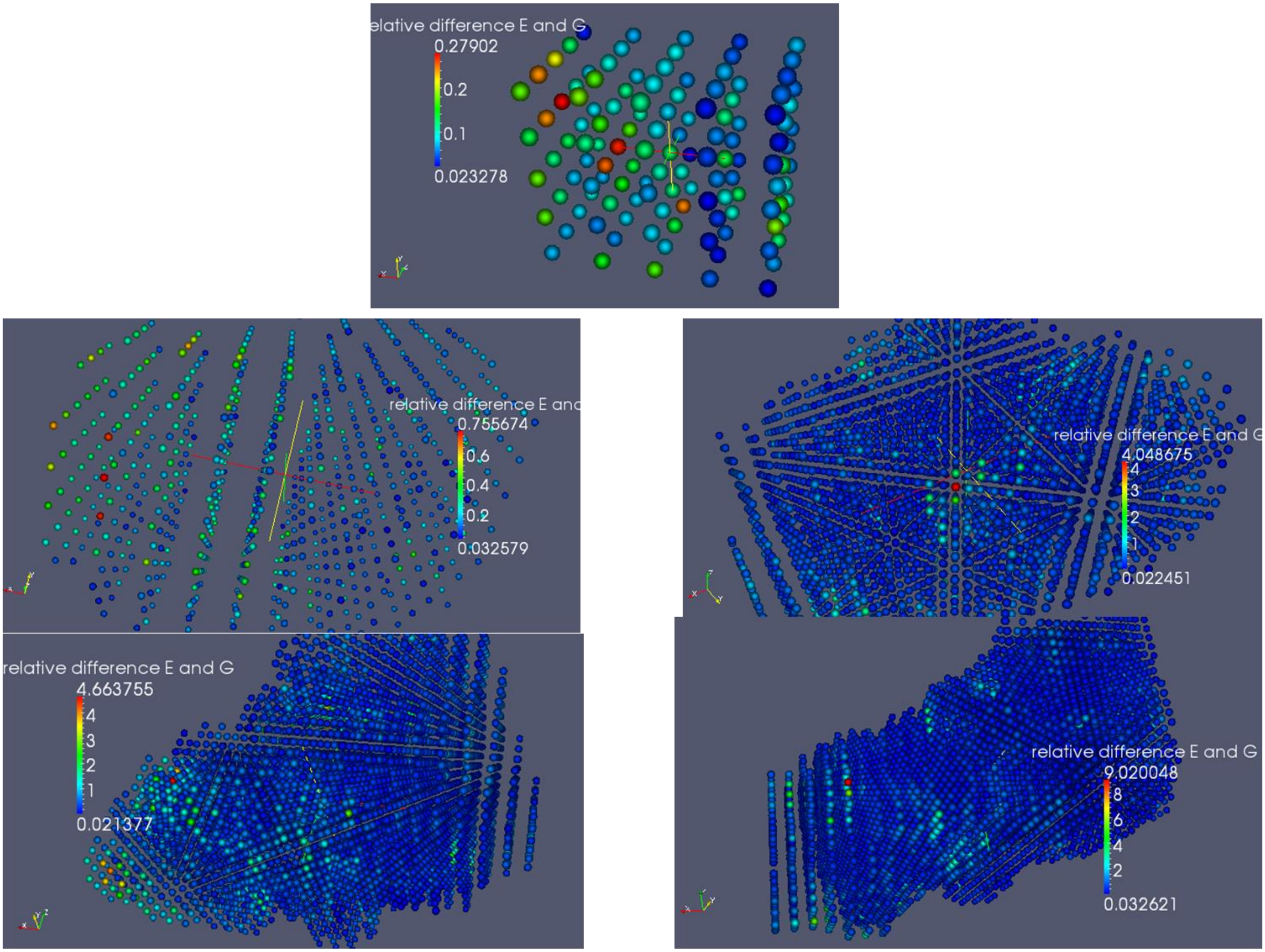

elative difference E and G
0.27902
0.2
0.1
0.023278
relative difference E anc
0.755674
0.6
0.4
0.2
0.032579
relative difference E and G
4.048675
4
3
2
1
0.022451
relative difference E and G
4.663755
4
3
2
1
0.021377
relative difference E and G
9.020048
8
6
4
2
0.032621

Figure 11: Ratio of the Frobenius norm of the difference of the linear strain and the Green strain to the Green strain norm. Here we present results for the 34th time interval with grids consisting of (going from top to bottom and from left to right) 5x5x5, 10x10x10, 20x20x20, 25x25x25 and 30x30x30 elements. Notice how as the grid becomes finer the highest relative difference (occurring at one or two points) becomes greater.

using the Frobenius norm) between the Linear and Green strains is greater the finer the grid used.

3) Moving Least Squares

3.1) Introduction to Moving Least Squares

We can actually improve our approximation by looking for approximating functions that have reproducibility beyond constants. A common approach is to use a least squares approximation. However, there are several problems with implementing such an approach here, namely: we do not have an initial guess as to what sort of function could be our underlying function generating the data over the whole domain; secondly, we are dealing with around 3000 atoms (data points) so the system will be incredibly overdetermined for small basis (i.e. basis of polynomials of low order). We can apply the ideas of a weighted least squares approximation locally. Hence, instead of dealing with one big linear system we deal with several smaller linear systems which individually are easier to solve, and we can generate a function which is a piecewise polynomial of lower order.

In general for the weighted least squares formulation we look to minimize the sum of the square of the difference between the approximated function and the original function at data sites, multiplied by the weight function at that site. Namely if we consider the space of polynomials of total degree m in d dimensions $\prod_m^d$ we want to minimize:

(9) $$\underset{f \in \prod_m^d}{min} \sum_{p=1}^{N_p} w(\mathbf{x} - \mathbf{x}_p)||\tilde{f}(\mathbf{x}_p) - u_p||$$

Where, as before, $\tilde{f}$ is the approximating function and u_p is the value of data to be approximated at data point p. Moreover we take $\tilde{f}$ to be of the form:

(8) $\tilde{f}(\boldsymbol{x}) = \boldsymbol{b}(\boldsymbol{x}) \cdot \boldsymbol{c}(\boldsymbol{x})$

Where $\boldsymbol{b}(\boldsymbol{x}) = [b_1(\boldsymbol{x}), b_2(\boldsymbol{x}), b_3(\boldsymbol{x}), \ldots, b_k(\boldsymbol{x})]$ is the basis vector (in this case for the polynomial basis), and **c**(x) is a vector of unknown coefficients that we wish to minimize (Nealen, 2004). Thus, taking derivatives with respect to the elements of **c** and minimizing we get:

(9) $$\boldsymbol{c}(\boldsymbol{x}) = \left[\sum_{\boldsymbol{p=1}}^{\boldsymbol{N_p}} \omega(\boldsymbol{x} - \boldsymbol{x_p})\, \boldsymbol{b}(\boldsymbol{x_p})\boldsymbol{b}(\boldsymbol{x_p})^{\boldsymbol{T}}\right]^{-1} \sum_{p=1}^{N_p} \omega(\boldsymbol{x} - \boldsymbol{x_p})\, \boldsymbol{b}(\boldsymbol{x_p}) u_p$$

Thus we can evaluate our function at any point in our domain by solving the linear system above for that point.

Concerning the moving least squares method there are two main choices we have to make for its implementation: choice of basis function and of weight functions. The choice of basis functions will determine the ability of our approximating function to reproduce the function that generated the data. On the other hand our choice of weight function determines the smoothness of our approximation (which we may want to be smoother than our original data). Indeed, the continuity of the approximation will be that of the weight function, even if the basis function has greater continuity than the weight function (Dolbow and Belytschko, 1998). Thus, even when the choice of basis involves functions with continuity of low order (e.g. a basis of polynomials of low order, such as a linear basis) the approximants can acquire arbitrary continuity by a suitable choice of weight functions.

Moreover, we observe that the Shepard's function can easily be recovered if we choose to use as our set of polynomials those of degree zero. So we can consider Shepard's function just a case of a moving least squares approximation.

For the purposes of implementing the moving least squares method we considered only polynomials as our basis functions. However, problems with conditioning arose quite quickly in the process as we tried using higher order polynomial basis. Indeed for second order polynomial basis the linear systems generated to approximate the values at most nodes had condition numbers greater than 10^{10}. These conditioning issues have been noted in the literature and a variety of strategies have been suggested to overcome this problem; including dealing with ill conditioned linear systems using singular value decomposition (Gossler, 2001); and by using a carefully chosen basis (Li and Liu, 2002). Nonetheless it is important that conditioning problems may also arise from the characteristics of the data.

For the present project we limited ourselves to the use of polynomials with the monomial basis. For three spatial variables (x,y,z) this is:

0-order b(x) = (1)

1st-order b(x) = (1,x,y,z)

2^{nd}-order b(x)= $(1,x,x^2,y,y^2,z,z^2)$

As mentioned above we only obtained reasonably well conditioned systems for the linear basis.

Concerning our choice of weight functions we considered cubic splines (figure 12) and thin plate splines (figure 13). Both of these weight functions have higher continuity than the linear basis employed previously. A significant difference between the two is that in the case of cubic splines the weight increases towards the center (i.e. the grid point) while for the thin plate splines the value of the weight is zero at the center and increases radially away from it. For both of these functions the support extended over four grid element lengths in every direction (as opposed to the case for the Shepard's function where we set the support to extend only two element lengths in every direction).

Cubic splines are given by:

$$\tilde{\omega}(r) = \begin{cases} \frac{2}{3} - 4r^2 + 4r^3 & , \; for\, r \leq \frac{1}{2} \\ \frac{4}{3} - 4r + 4r^2 - \frac{4}{3}r^3 & , \; for \frac{1}{2} < r \leq 1 \\ 0 & , \; for\, r > 1 \end{cases}$$

$$\omega(\boldsymbol{x} - \boldsymbol{x_g}) = \tilde{\omega}(r_1)\tilde{\omega}(r_2)\tilde{\omega}(r_3)$$

Where

$$r_1 = \frac{|x_p - x_g|}{2\Delta x} \quad , r_2 = \frac{|y_p - y_g|}{2\Delta y} \quad , r_3 = \frac{|z_p - z_g|}{2\Delta z}$$

Whereas thin plate splines are given by :

$\omega(r) = r^2 \ln{(r+1)}$, where $r = \sqrt{r_1^2 + r_2^2 + r_3^2}$

In figure 12 we plot the cubic spline weight function in two dimensions and in figure 13 we plot the thin plate spline in two dimensions.

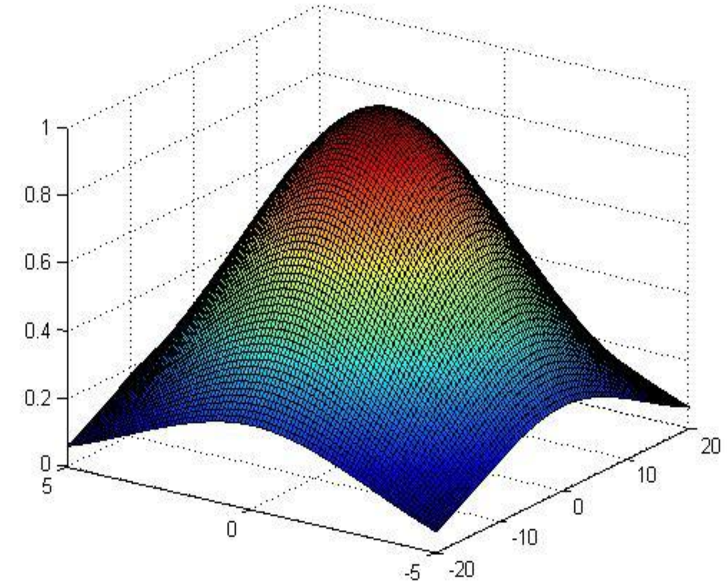


Figure 12: Cubic spline weight function with rectangular support centered at the origin.

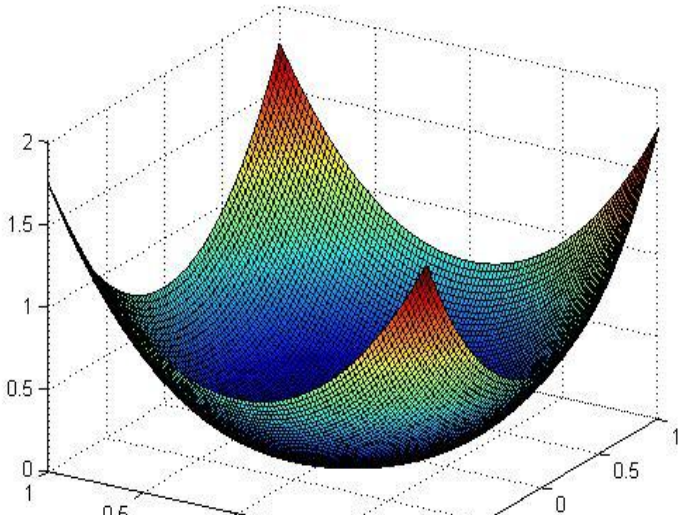


Figure 13: Thin plate spline weight function centered at the origin with unit square support centered at the origin. Note how the value of this weight function increases towards the edges of the support.

We consider some examples in one and two dimensions. As before we use a test function and compare the MLS approximations with the values of the test function at our grid points (figure 14).

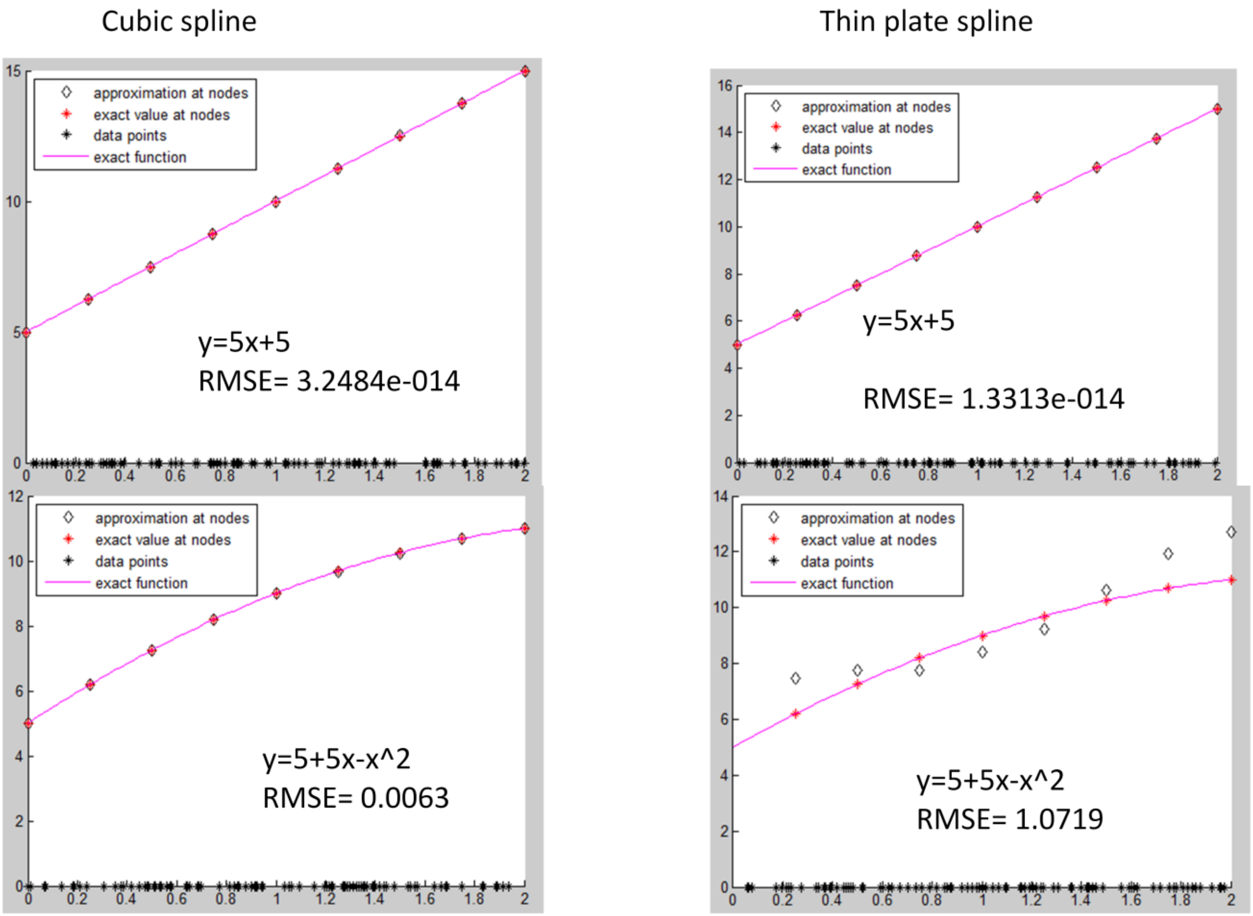


Figure 14: Examples of known functions being approximated using MLS approximants with polynomial basis of degree one. (Left) We approximate using a cubic spline as a weight function, we get a good approximation even in the case that the function to be approximated is not linear. (Right) We approximate using a thin plate spline as a weight function; notice how in the case of the quadratic function the error for this approximation grows dramatically. In both cases we reproduce linear functions (RMSE is near machine precision). These examples were done using values N_g=8 and N_p = 100.

We notice that unless the approximated function is almost linear the approximation using thin plate splines produces large errors. Thus approximations using such weight functions may provide indication as to how much the function to be approximated deviates from a linear function. We see similar results for the 2D cases (figures 15 and 16). In our case, if the function to be approximated is not close to linear the thin plate spline approximants will not yield good results.

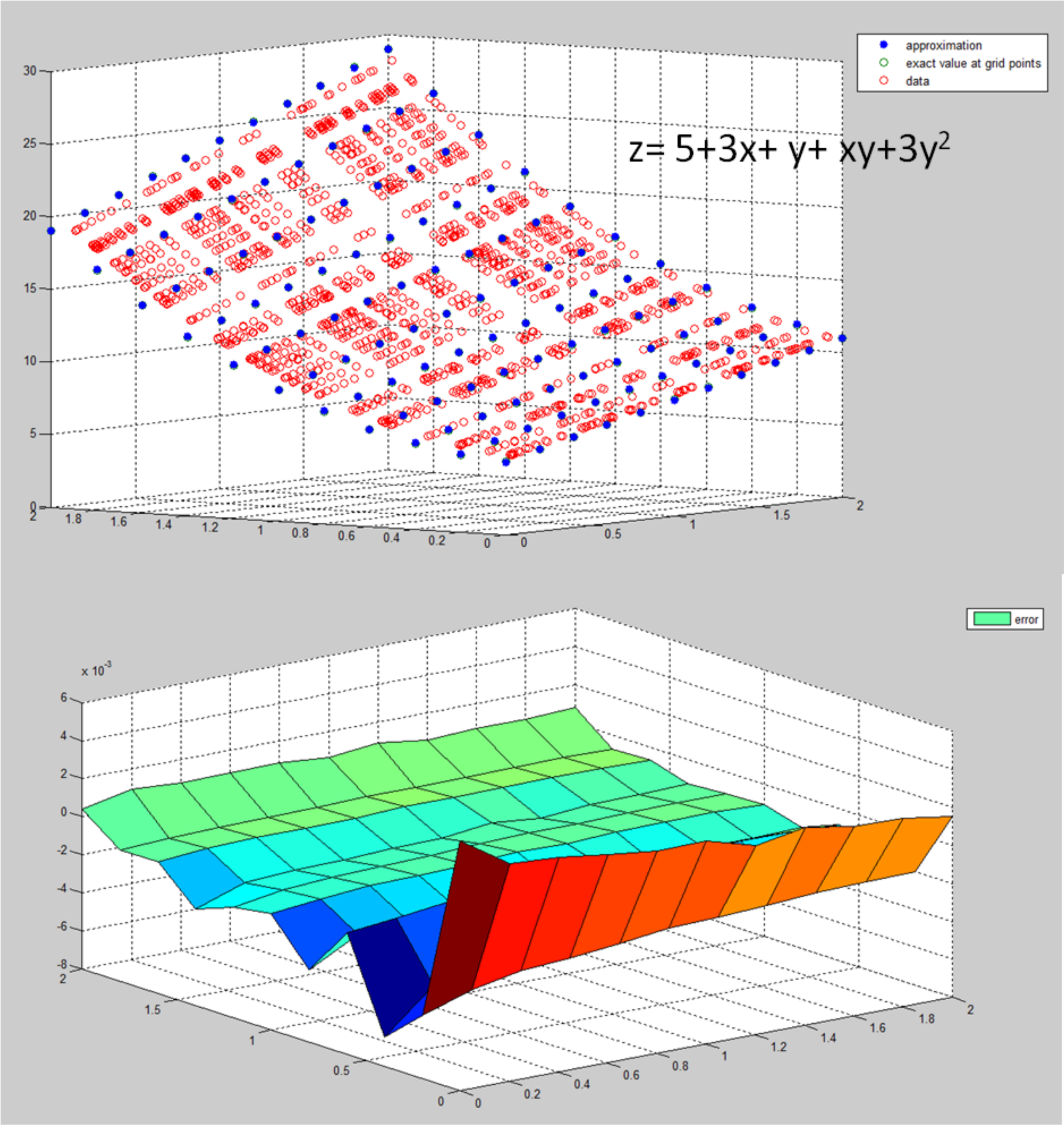


Figure 15: Example of known functions being approximated using MLS approximants with polynomial basis of degree one. We approximate using a cubic spline as a weight function, we get a good approximation even in the case that the function to be approximated is not linear. Top: plot showing the approximated values in our grid (blue) the exact values at the grid points (green) and the data approximated (red). Bottom: relative error for our approximation. These examples were done using values N_p = 1600 and N_g =N_{gx}*N_{gy} =10*10.

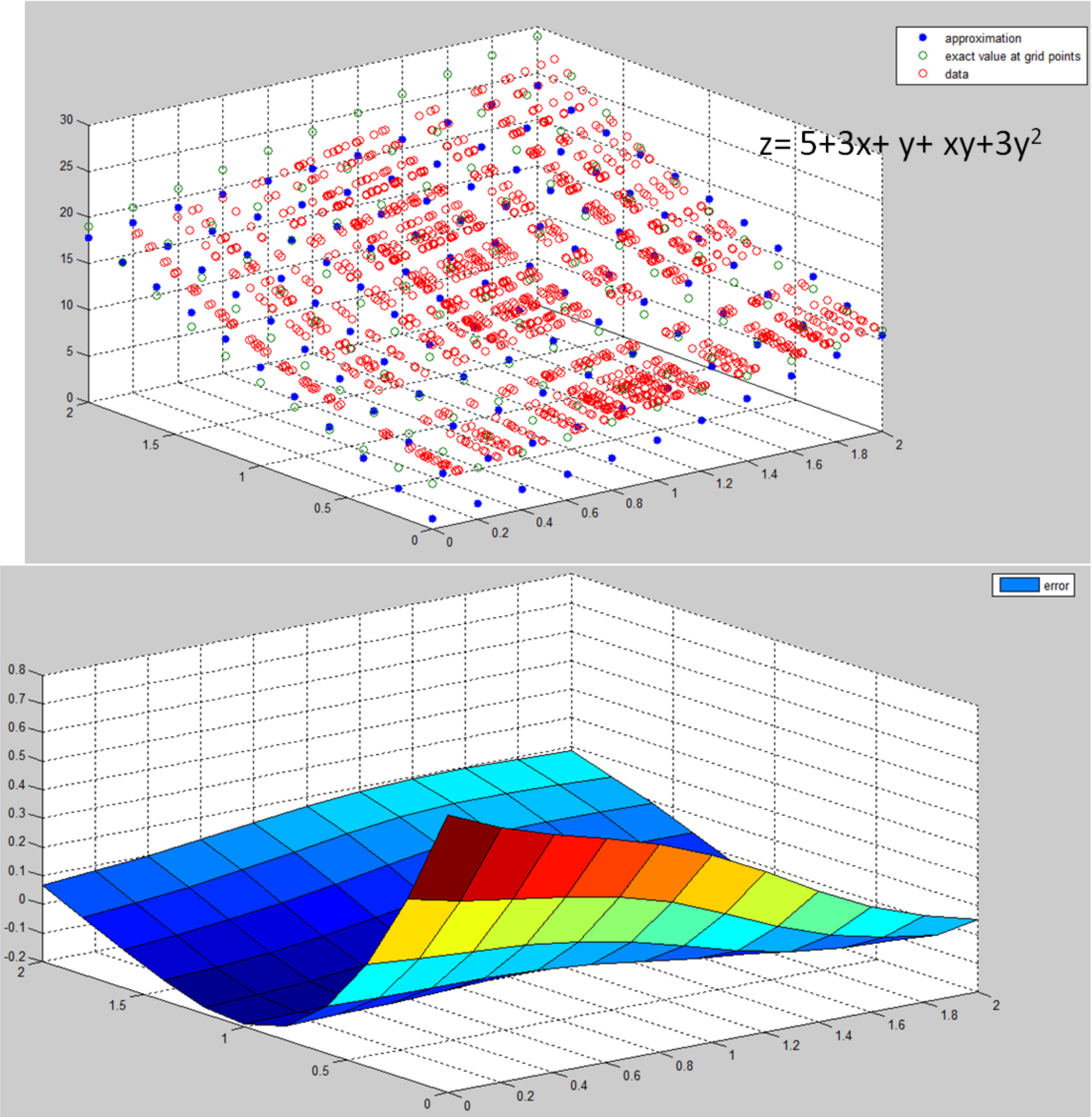
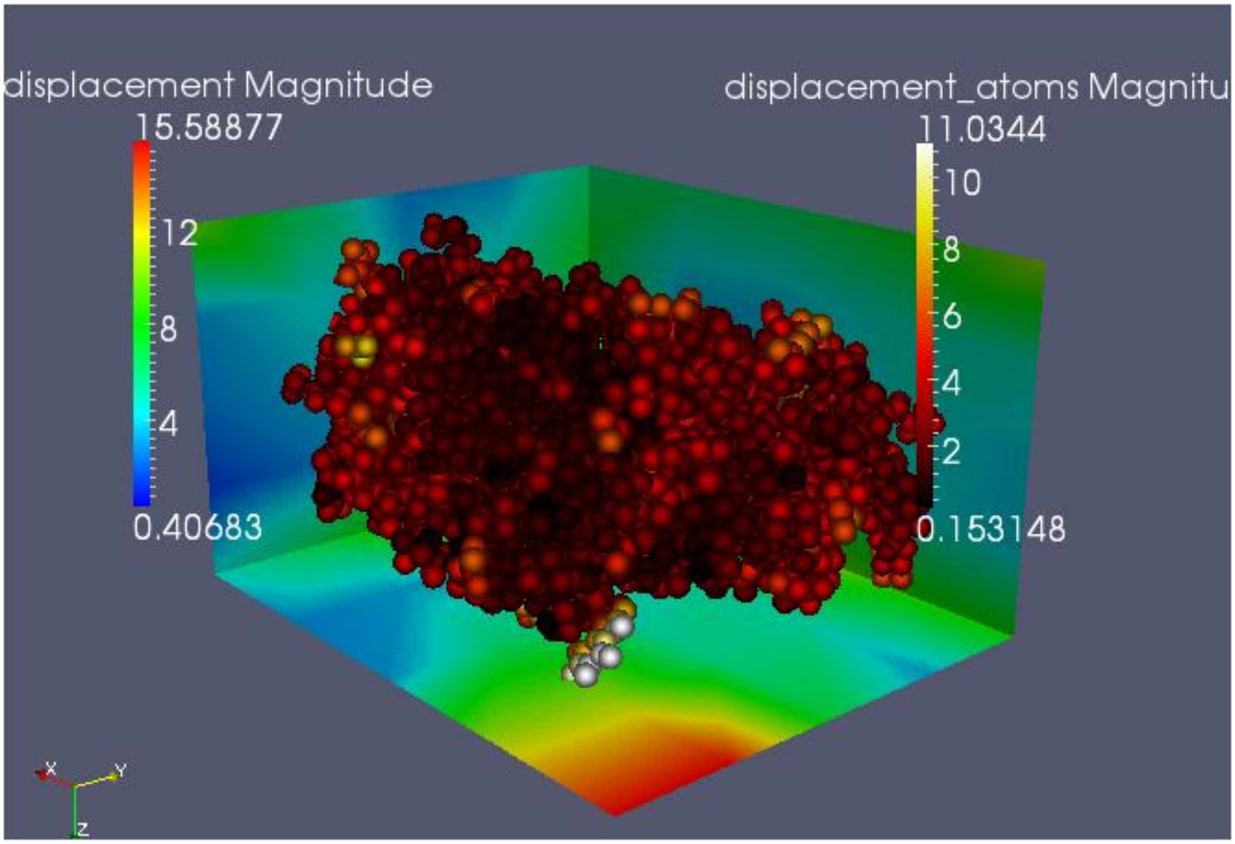


Figure 16: We approximate using a thin plate spline as a weight function; notice that the relative error for this approximation is significantly larger than the relative error for the approximation that uses cubic splines. Top: plot showing the approximated values in our grid (blue) the exact values at the grid points (green) and the data approximated (red). Bottom: relative error for our approximation. These examples were done using values N_p = 1600 and N_g =N_{gx}*N_{gy} =10*10.

3.2) MLS Application to Molecular Dynamics Simulation Data

Again we consider dividing the domain spanned by HIV-PR by elements with a size of an order 10 times that of inter-atomic distances. And we observe good correspondence between regions of high linear strain in our grid and high deformation in the molecule (figures 17) for approximations where a cubic spline was used as a weight function. This indicates that to a first degree, regions of high deformation can be identified and characterized at scales much coarser than the inter-atomic scale.

(a)



(b)

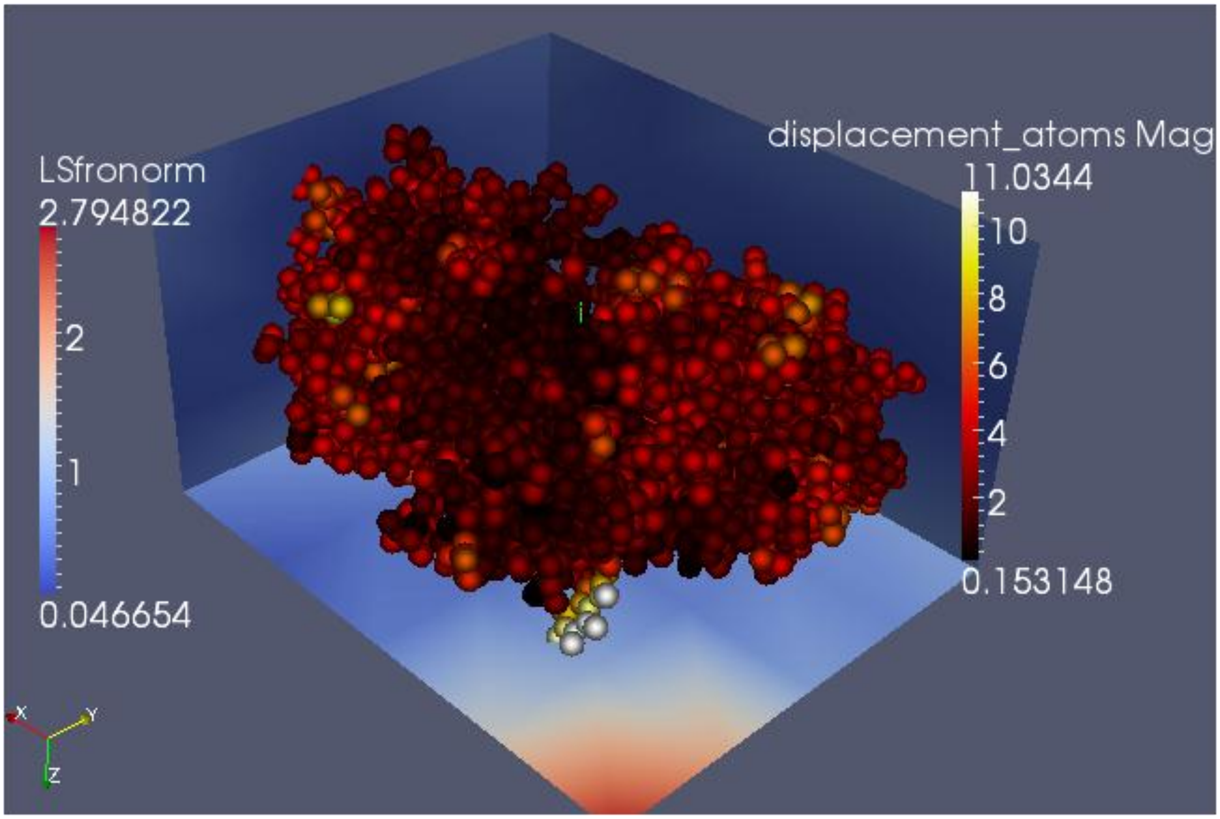


Figure 17: Results using cubic splines weight functions, linear polynomial basis functions, a grid of 4X4X4 cells, for the 9th time step. Atom locations are represented by the spheres and the surface is a result of a coloring interpolation performed by paraview based on the results for our grid. (a) Magnitude of displacement in the grid vs magnitude of displacement at the atom's locations. (b) Frobenius norm of the linear strain (in the background) compared to values of displacement at the atoms' locations. Here the results for displacement and deformation in our grid match the results for the atoms' positions.

Nonetheless, we notice that there is not a good correspondence for approximations for which thin plate splines were used (figure 18). Which probably indicates that the underlying function that generates the data for the atoms' displacement deviates from linear behavior at length scales of roughly ten times the inter-atomic distance.

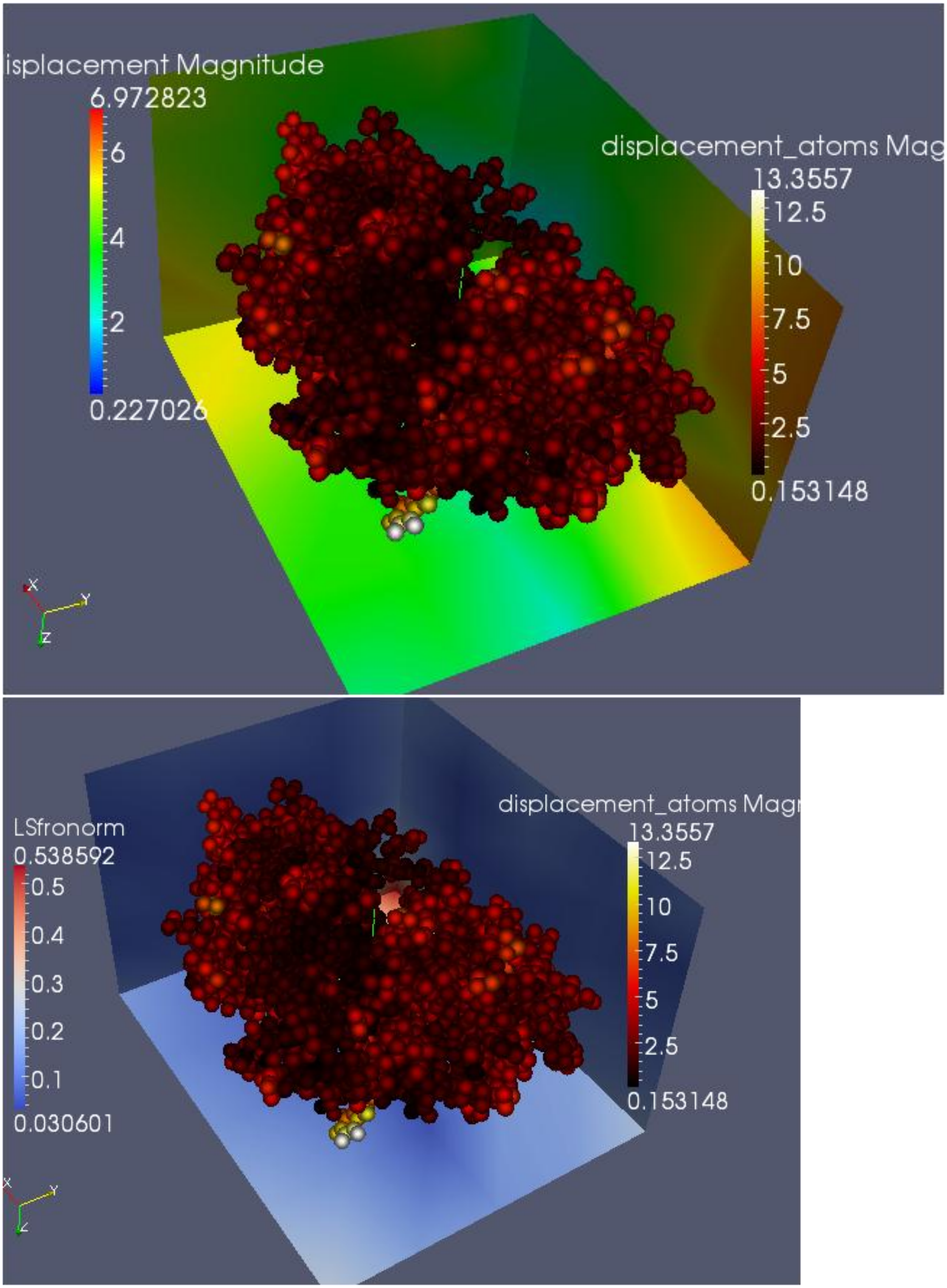


Figure 18: Figure 19: Results using thin plate spline weight functions, linear polynomial basis functions, a grid of 4X4X4 cells, for the 9th time step. Atom locations are represented by the spheres and the surface is a result of a coloring interpolation performed by paraview based on the results for our grid. (Top) Magnitude of displacement in the grid vs. magnitude of displacement at the atom's locations. (Bottom) Frobenius norm of the linear strain (in the background) compared to values of displacement at the atoms' locations. Here the results for displacement and deformation in our grid are not in good correspondence the results for the atoms' positions.

4) Concluding remarks

We observed that the results in our grids reproduce the behavior of the data in a coarser scale. We observe regions of large deformations and displacements at the flaps and edges for all of our simulations. Such features are retained even when we consider scales (grid element dimensions) that are comparable to ten times the average inter-atomic distance of HIV-PR atoms. Thus we can see that basic deformation features occurring in HIV-PR dynamics may be studied at larger scales than the atomic scale.

Although we faced a number of problems with conditioning this problem may be overcome by a more adequate choice of basis functions. Also, there is an interesting possibility for exploring the use of a non-uniform grid, for example where the grid elements are shaped according to molecular structures found in the crystal structure of HIV-PR. Information for mechanical parameters maybe included by using information about the strength of chemical bonds and chemical composition, which could serve to parameterize rigidity and density respectively. Moreover, shorter time scales may be considered in the future to explore processes that may not be seen at the time scale set by the time steps of our data.

References

Belytschko, T. et al “Meshless Methods: An Overview and Recent Developments.” 1996.

Brik, A and Wong, C. HIV-1 protease: mechanism and drug discovery , Org. Biomol. Chem., 2003,1,5-14

Carlson, H.A. and McCammon, J.A. Accommodating Protein Flexibility in Computational Drug Design, MOLECULAR PHARMACOLOGY, 57:213–218 (2000).

Dolbow, J. and Belytschko, T. “An introduction to programming the meshless Free Element Galerkin Method.” Archives in Computational Methods in Engineering, Vol. 5, 1998.

Fasshauer, G.E. Meshfree Approximation Methods with Matlab. World Scientific Publishing, 2007.

Gonzalez, O. and Stuart A.M., A first course in continuum mechanics, Cambridge, UK ; New York : Cambridge University Press, 2008.

Gossler, A. Moving Least-Squares: A Numerical Differentiation Method for Irregularly Spaced Calculation Points, SANDIA REPORT, SAND2001-1669, Printed June 2001.

Hornak, V. and Simmerling, C. “Targeting structural flexibility in HIV-1 protease inhibitor binding.” Drug Discovery Today Volume 12, Numbers 3/4 February 2007.

Lancaster, P. and Salkauskas, K. “Surfaces Generated by Moving Least Squares Methods.” Mathematics of Computation, volume 37, number 155, JULY 1981

Li, S. and Liu, W.K. Meshfree and particle methods and their applications. Appl Mech Rev vol 55, no 1, January 2002

Nealen , A. An As-Short-As-Possible Introduction to the Least Squares, Weighted Least Squares and Moving Least Squares Methods for Scattered Data Approximation and Interpolation, 2004. Found at http://www.nealen.com/projects/

ParaView User's Guide (v3.10) – found at http://paraview.org/paraview/help/download/ParaView%20User%27s%20Guide%20v3.10.pdf

Pokorná et al, Current and Novel Inhibitors of HIV Protease, *Viruses* 2009, *1*, 1209-1239; doi:10.3390/v1031209

Shepard, D. A two-dimensional interpolation for irregularly-spaced data. Proceedings-1968 ACM National Conference.

Wong, C. F. and McCammon J. A., PROTEIN FLEXIBILITY AND COMPUTER-AIDED DRUG DESIGN, Annu. Rev. Pharmacol. Toxicol. 2003. 43:31–45

Weinberg, K. Lecture Notes, Chapter 2 : Kinematics of Deformation, found at: http://mechanik.tu-berlin.de/weinberg/Lehre/fem2/Chapter2.pdf

Appendix A

The approximations performed for this project used as data the results of molecular dynamics simulations courtesy of Carlos Simmerling from the Department of Chemistry at the State University of New York. The values for the atomic positions at regular time steps of the order of 10 micro seconds were given as coordinates in a system where the unit is one angstrom. Such information was included in 2021 protein data base (.pdb) files. The atoms' coordinates were read into Matlab and they were transformed into the centroid reference frame. Such transformation takes the origin (centroid's position) to be equal to the average position of atoms, and then subtracts that position vector from the position vectors for the atoms. For the coordinate transformation the following code was implemented:

```
function [R,X,Y,Z] = Ctrans(x,y,z)
%[R,X,Y,Z] = Ctrans(m,x,y,z)
% This function gives the centroid position R as a vector and
% transforms the coordinates (x,y,z) to the centroid coordinates
% (X,Y,Z).
% Inputs:   x,y,z : column vectors containing the coordinates for all the
%           particles.
% Outputs:  R:  a vector containing the position of the center of mass (in
%               original system coordinates).
%           X,Y,Z:  vectors containing the coordinates of the particles'
%                   postions in the CM frame.

M = length(x);
Rx = (1/M)*sum(x);
Ry = (1/M)*sum(y);
Rz = (1/M)*sum(z);

R = [Rx Ry Rz];

X = x-Rx;
Y = y-Ry;
Z = z-Rz;

end
```

Thus the atoms' coordinates in the new coordinate system were stored as Matlab variables in .mat files.

We used the open source visualization application Paraview developed by Kitware, Sandia National Labs and CSimSoft to view our results. This application was capable of turning the vector components into vector data that can be visualized as arrows in a three dimensional representation. Matlab results were turned into visualization toolkit files (.vtk) files that could be opened in paraview as well as comma separated value files (.csv) for which paraview could read individual columns of values and then plot them as plots with coordinates specified by three of the value columns (see figure 15).

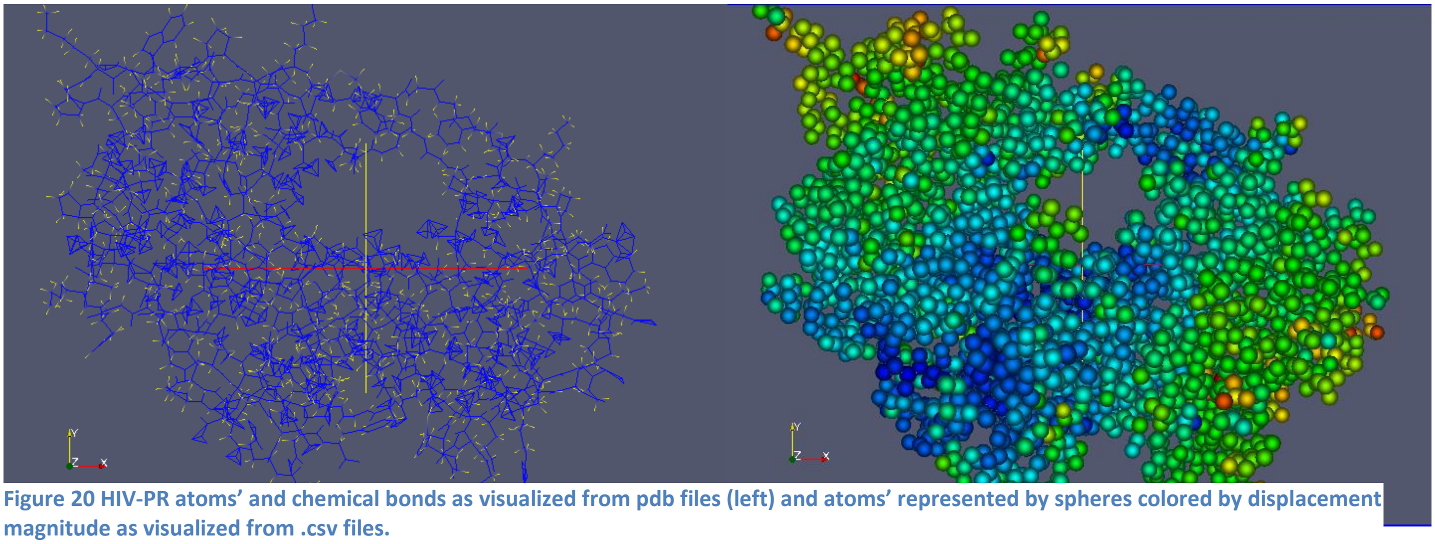


Figure 20 HIV-PR atoms' and chemical bonds as visualized from pdb files (left) and atoms' represented by spheres colored by displacement magnitude as visualized from .csv files.

We also used the Matlab class written by Michael Mallon of the University of Queensland Australia to write our vtk files. The package can be found at: http://www.rcc.uq.edu.au/download/VtkWriter/

Appendix B: Strain measures

Strain can be thought of as being a way to describe deformation undergone by a solid. Consider a solid occupying a subset B of Euclidean space $\mathbb{R}^3$;

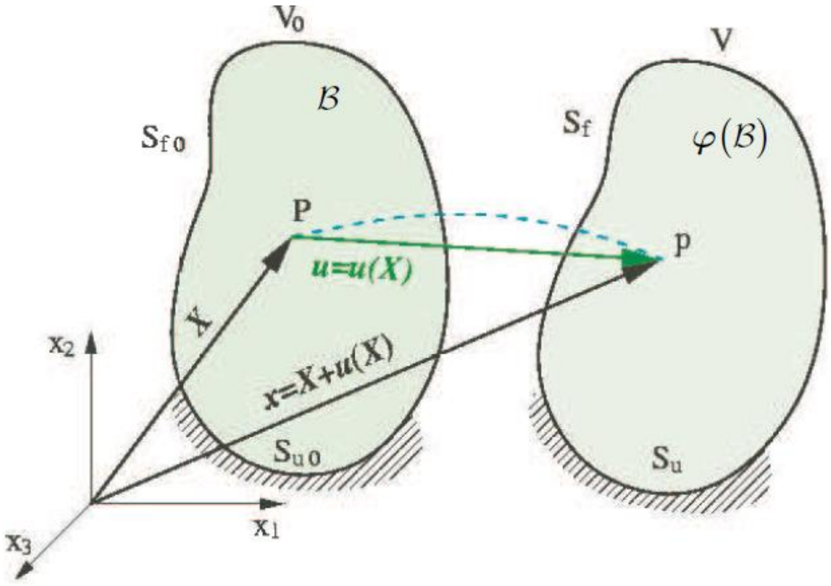


Figure 21: taken from http://mechanik.tu-berlin.de/weinberg/Lehre/fem2/Chapter2.pdf lecture notes

And let a particle's original position be given by **X**, an element of B. At a later time let the body occupy a subset B' of R^3, and the particle's position be given by **x**, an element of B'. We have a map φ from B to B' so that $\varphi(\boldsymbol{X}) = \boldsymbol{x}$. We define displacement by

(B1) $U(\boldsymbol{X}) = \varphi(\boldsymbol{X}) - \boldsymbol{X} = \boldsymbol{x} - \boldsymbol{X}$

Clearly such a displacement includes rigid body rotations and translations as well as deformations. We would like a quantity which is not affected by such rigid body motions, Define:

$F = \nabla_0 \varphi(\boldsymbol{X})$ as the deformation gradient.

The ∇_0 operator represents derivatives with respect to **X**. From B1 we see that:

$$F = \nabla_0 \varphi(\boldsymbol{X}) = \nabla_0 U(\boldsymbol{X}) + I$$

The deformation gradient gives us information about how small segments deform. Let **X** and **X**+d**X** be two points in the original configuration. In the deformed configuration $\varphi(\boldsymbol{X}) = \boldsymbol{x}, \boldsymbol{x} + \boldsymbol{dx} = \varphi(\boldsymbol{X} + \boldsymbol{dX})$**.** For infinitesimal d**X**

$$\boldsymbol{x} + \boldsymbol{dx} = \varphi(\boldsymbol{X} + \boldsymbol{dX}) = \varphi(\boldsymbol{X}) + \boldsymbol{\nabla}_0 \varphi(\boldsymbol{X}) \cdot \boldsymbol{dX} \rightarrow \boldsymbol{dx} = \boldsymbol{F} \cdot \boldsymbol{dX}$$

So a segment of length d**X** in the original configuration transforms to a segment of length d**x** in the deformed configuration under the action of F.

If we examine the square of the length of the segment d**x**, d**x**^{T}d**x**, we see

$$d\boldsymbol{x}^T d\boldsymbol{x} = d\boldsymbol{X}^T \boldsymbol{F}^T \boldsymbol{F} d\boldsymbol{X}$$

Define the Cauchy-Green Strain Tensor $C = F^T F$

$$C = F^T F = \left(I + \nabla_0 U(X)\right)^T \left(I + \nabla_0 U(X)\right) = I + \nabla_0 U(X)^T + \nabla_0 U(X) + \nabla_0 U(X)^T \nabla_0 U(X)$$

It can be shown (Gonzalez and Stuart, 2008) by the polar decomposition theorem that the deformation gradient tensor can be expressed as the product of an orthogonal tensor (R) and a positive definite symmetric tensor (U).

F = RU

Where R is a tensor that describes a solid body rotation and U describes stretch.

Hence we observe that

$C = F^TF = (RU)^TRU = U^TR^TRU = U^TU=U^2$

since R is orthogonal.

Moreover the infinitesimal strain tensor can be defined as

$$E = sym(\nabla_0 \boldsymbol{U}) = \frac{1}{2}(\nabla_0 \boldsymbol{U} + \nabla_0 \boldsymbol{U}^{\mathrm{T}})$$

Let

$G = \frac{1}{2}(C - I)$ The Green strain tensor

So we have

$$G - E = \frac{1}{2}\nabla_0 \mathbf{U}^{\mathrm{T}} \nabla_0 \mathbf{U}$$

Thus as the deformation approaches zero the difference between the infinitesimal strain tensor and the Green strain tensor approaches zero.

Appendix C: Coding a Moving Least Squares implementation in Matlab

Coding a Moving Least Squares implementation in Matlab involves looping over the grid elements in order to setup the linear system to be solved for each grid point. Hence This also involves finding which data points are within a grid point's weight function support. This can be achieved either via logical indexing (i.e. by searching which set of data points is inside a cube of a given length around a grid point) or by specifying in advance indices for the grid elements and grid points (thus data points in grid element I contribute to the grid points in that element's corners) . The latter approach was used to implement Shepard's method while the former approach was used to implement the moving least squares method for higher order basis (see attached codes).