LA-UR-17-24975
Approved for public release; distribution is unlimited.

Title: A Simple Introduction to Moving Least Squares and Local Regression Estimation

Author(s): Garimella, Rao Veerabhadra

Intended for: Report

Issued: 2017-06-22 (rev.1)

A Simple Introduction to Moving Least Squares and Local Regression Estimation

Rao V. Garimella (rao@lanl.gov)
T-5, Los Alamos National Laboratory

Jun 20, 2017

LA-UR-17-24975

Abstract

In this brief note, a highly simplified introduction to esimating functions over a set of particles is presented. The note starts from Global Least Squares fitting, going on to Moving Least Squares estimation (MLS) and finally, Local Regression Estimation (LRE).

1 Introduction

Here we consider the problem of estimating a general function over a domain given a set of pointwise values over that domain. Such an estimation is useful for numerical solution of PDEs over a domain, surface reconstruction from scanned data or understanding the structure of collected data. We will describe a series of techniques that lead up to a powerful method called Local Regression Estimator that has many desirable properties.

2 Global Least Squares Estimation

Consider the problem of fitting a function to a data set in 1D. The points are given by $x_i, i = 1, N$ and the function values at these points by $u(x_i) = u_i, i = 1, N$. If we assume that the data can be approximately represented by a global polynomial function then we can write:

$$\begin{aligned} u^h(x) &= \sum_j^m b_j(x) c_j \\ &= \mathbf{b}^T(x)\mathbf{c} \end{aligned} \tag{1}$$

where, u^h is the approximation to the true function $u(x)$, $\mathbf{b}(x)$ is a *basis* vector containing monomials up to degree m defined as $\begin{bmatrix}1 & x & x^2 & \dots & x^m\end{bmatrix}^T$, $\mathbf{c}$ is a vector of *constant* coefficients defined as $\begin{bmatrix}c_0 & c_1 & c_2 & \dots & c_m\end{bmatrix}^T$.

To solve for the unknown coefficients of the polynomial fit, we construct the following objective function:

$$\begin{aligned}\mathcal{R}^{GLS} &= \sum_i^N \left(u^h(x_i) - u_i\right)^2 \\ &= \sum_i^N \left(\sum_j^m b_j(x_i)c_j - u_i\right)^2\end{aligned} \tag{2}$$

and minimize it with respect to the coefficents $\mathbf{c}$. Setting $\partial\mathcal{R}^{GLS}/\partial c_k = 0, k = 1, m$ to obtain the extrema, we get:

$$\begin{aligned}\sum_i^N 2\left(b_k(x_i)\left(\sum_j^m b_j(x_i)c_j - u_i\right)\right) = 0, \qquad \text{or,} \\ \sum_i^N \left(b_k(x_i)\sum_j^m b_j(x_i)c_j\right) = \sum_i^N b_k(x_i)u_i\end{aligned} \tag{3}$$

We can then write the m normal equations as:

$$\begin{gathered}\begin{bmatrix}b_1(x_1) & b_1(x_2) & \dots & b_1(x_N)\end{bmatrix}\begin{bmatrix}b_1(x_1) & b_2(x_1) & \dots & b_m(x_1)\\ b_1(x_2) & b_2(x_2) & \dots & b_m(x_2)\\ & \vdots & & \\ b_1(x_N) & b_2(x_N) & \dots & b_m(x_N)\end{bmatrix}\begin{bmatrix}c_1\\c_2\\ \vdots\\ c_m\end{bmatrix} = \begin{bmatrix}b_1(x_1) & b_1(x_2) & \dots & b_1(x_N)\end{bmatrix}\begin{bmatrix}u_1\\u_2\\ \vdots\\ u_N\end{bmatrix}\\ \begin{bmatrix}b_2(x_1) & b_2(x_2) & \dots & b_2(x_N)\end{bmatrix}\begin{bmatrix}b_1(x_1) & b_2(x_1) & \dots & b_m(x_1)\\ b_1(x_2) & b_2(x_2) & \dots & b_m(x_2)\\ & \vdots & & \\ b_1(x_N) & b_2(x_N) & \dots & b_m(x_N)\end{bmatrix}\begin{bmatrix}c_1\\c_2\\ \vdots\\ c_m\end{bmatrix} = \begin{bmatrix}b_2(x_1) & b_2(x_2) & \dots & b_2(x_N)\end{bmatrix}\begin{bmatrix}u_1\\u_2\\ \vdots\\ u_N\end{bmatrix}\\ \vdots\\ \begin{bmatrix}b_m(x_1) & b_m(x_2) & \dots & b_m(x_N)\end{bmatrix}\begin{bmatrix}b_1(x_1) & b_2(x_1) & \dots & b_m(x_1)\\ b_1(x_2) & b_2(x_2) & \dots & b_m(x_2)\\ & \vdots & & \\ b_1(x_N) & b_2(x_N) & \dots & b_m(x_N)\end{bmatrix}\begin{bmatrix}c_1\\c_2\\ \vdots\\ c_m\end{bmatrix} = \begin{bmatrix}b_m(x_1) & b_m(x_2) & \dots & b_m(x_N)\end{bmatrix}\begin{bmatrix}u_1\\u_2\\ \vdots\\ u_N\end{bmatrix}\end{gathered} \tag{4}$$

in matrix form as:

$$\mathbf{B}^T\mathbf{B}\mathbf{c} = \mathbf{B}^T\mathbf{u} \tag{5}$$

where $\mathbf{B}$ is an $m \times N$ matrix defined as:

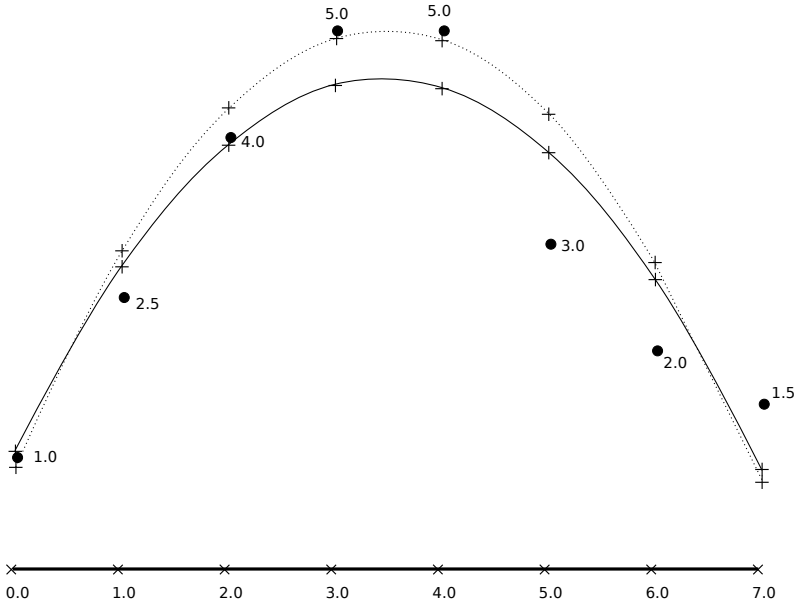

Figure 1: Global Least Squares (solid curve) and Weighted Global Least Squares (dashed curve) fit for data represented by the solid circles. The fit is a quadratic fit. The weighting used for the dashed curve is 1.0 every where except at $x = 3.0$ and $x = 4.0$ where it is 10.0

$$\mathbf{B} = \begin{bmatrix} b_1(x_1) & b_2(x_1) & \dots & b_m(x_1) \\ b_1(x_2) & b_2(x_2) & \dots & b_m(x_2) \\ & \cdot & & \\ & \cdot & & \\ & \cdot & & \\ b_1(x_N) & b_2(x_N) & \dots & b_m(x_N) \end{bmatrix} \tag{6}$$

This matrix equation can be solved directly for the coefficient vector $\mathbf{c}$ as:

$$\mathbf{c} = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{u} \tag{7}$$

or in case of large systems using an iterative method.

Having obtained the coefficients $\mathbf{c}$, we can then compute the value of the function at any point x in the domain using the equation for u^h. This analysis is substantially unchanged in higher dimensions. An example of a global least squares fit is shown in Figure 1

3 Weighted Global Least Squares fit

We can assign each data value a weight w to use in the least squares fit, so that we write the objective function to be minimized as:

$$\begin{aligned}\mathcal{R}^{WGLS} &= \sum_i^N w_i \left(u^h(x_i) - u_i\right)^2 \\ &= \sum_i^N w_i \left(\sum_j^m b_j(x_i)c_j - u_i\right)^2\end{aligned} \tag{8}$$

Setting $\partial\mathcal{R}^{WGLS}/\partial c_k = 0, k = 1, m$, we get the normal equations as:

$$\begin{aligned}2\sum_i^N \left(b_k(x_i)w_i\left(\sum_j^m b_j(x_i)c_j - u_i\right)\right) = 0, \qquad &\text{or,} \\ \sum_i^N \left(b_k(x_i)w_i\sum_j^m b_j(x_i)c_j\right) = \sum_i^N b_k(x_i)w_i u_i \\ \mathbf{B}^T\mathbf{W}\mathbf{B}\mathbf{c} = \mathbf{B}^T\mathbf{W}\mathbf{u}\end{aligned} \tag{9}$$

where $\mathbf{W}$ is the *weight* matrix defined as:

$$\begin{bmatrix} W_1 & 0 & \dots & 0 \\ 0 & W_2 & \dots & 0 \\ & & \cdot & \\ & & \cdot & \\ & & \cdot & \\ 0 & 0 & \dots & W_N \end{bmatrix} \tag{10}$$

The solution of the normal equations is then given as:

$$\mathbf{c} = (\mathbf{B}^T\mathbf{W}\mathbf{B})^{-1}\mathbf{B}\mathbf{W}\mathbf{u} \tag{11}$$

The matrix $\mathbf{B}^T\mathbf{W}\mathbf{B}$, is called a *moment matrix*. An example of the weighted global least squares fit is shown in Figure 1.

4 Weighted Local Least Squares

In global least squares fitting, we assumed that the function represented by the data can be accurately captured by a single polynomial spanning the entire domain. For large, complex data sets, however, this would require us to fit a polynomial of an impractically high order and even then it may not capture all the features of the data. So instead of a global solution, we can attempt to gain a better picture of the solution by fitting lower order polynomials to each data point (x_p, u_p) and its "neighbors." Thus there are N least squares fits u_p^h, each one approximating the solution at point x_p and the coefficient vector c is different at each point. *Note that unlike other methods discussed here this is not a well-established method and is not in common use; instead,*

the method is being presented merely as a stepping stone on our way to understanding Moving Least Squares in the next section.

We now write the approximate solution at point x_p formally as:

$$\begin{aligned} u_p^h(x) &= \sum_j^m b_j(x) c_{pj} \\ &= \mathbf{b}^T(x)\mathbf{c}_p \end{aligned} \tag{12}$$

where u_p^h is shorthand for $u^h|_{x_p}$ and c_{pj} for $c_j|_{x_p}$. Since this is a low order polynomial, we do not expect to use this approximation at points "far away" from x_p. We then write the *local* objective function at point x_p as:

$$\begin{aligned} \mathcal{R}_p^{WLLS} &= \sum_i^N w_{pi} \left(u_p^h(x_i) - u_i \right)^2 \\ &= \sum_i^N w_{pi} \left(\sum_j^m b_j(x_i) c_{pj} - u_i \right)^2 \end{aligned} \tag{13}$$

to be minimized with respect to $\mathbf{c}$. As with variables u_p^h and c_{pj}, we use $\mathcal{R}_p^{WLLS}$ as shorthand for $\mathcal{R}^{WLLS}|_{x_p}$ and w_{pi} for $w_i|_{x_p}$. Note that in addition to the coefficients, the weighting of the data is different for the local objective function at each point x_p. This is because we typically will choose a low order polynomial (linear, quadratic, etc.) to fit locally and we would not want data from points far from the current point to exert an influence on the local solution. Thus the weights are non-zero for data points in the local neighborhood and are zero for points outside.

Then the the normal equations at point x_i become:

$$\begin{aligned} \sum_i^N \left(b_k(x_i) w_{pi} \sum_j^m c_{pj} b_j(x_i) \right) &= \sum_i^N b_k(x_i) w_{pi} u_i \\ \mathbf{B}^T \mathbf{W}_p \mathbf{B} \mathbf{c}_p &= \mathbf{B}^T \mathbf{W}_p \mathbf{u} \end{aligned} \tag{14}$$

which is solved in the usual way.

Unlike the Weighted Global Least Squares fit, there isn't necessarily only one approximate solution at any point x in the domain since the neighborhoods of the local solutions overlap. Then the first choice we have for evaluating $u^h(x)$ is to use the local solution u_p^h at the point x_p closest to point x. The other method could be to use a partition of unity that pulls in all approximate solutions that may be active over x like so:

$$u^h(x) = \frac{1}{n} \sum_i^n u_i^h \tag{15}$$

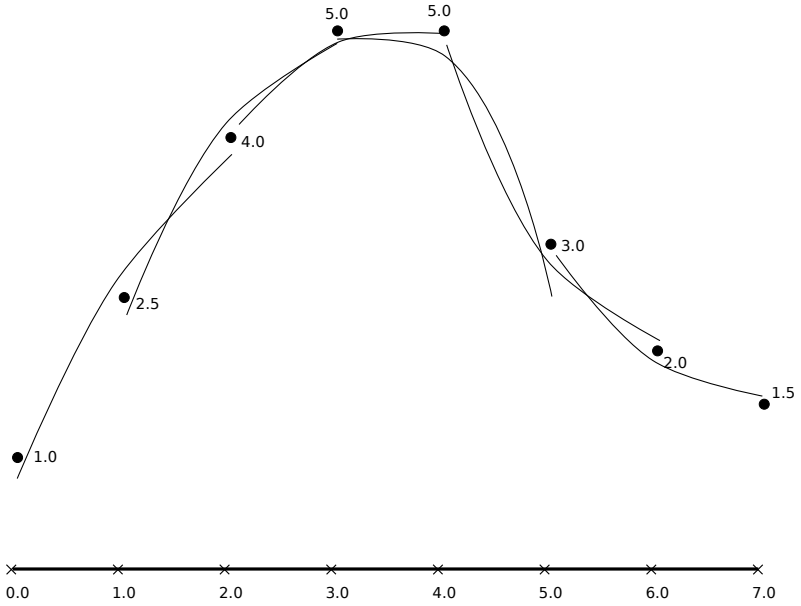

Figure 2: Weighted Local Least Squares fit for given data. Note the overlapping approximations at most points

where n is the number of solutions that are non-zero at x. An example of what quadratic functions fitted to each point and its neighbors might look like is shown in Figure 2.

5 Moving Least Squares

In the Moving Least Squares method (MLS), we build upon the notion of the Weighted Local Least Squares and extend it to build a local solution at *any* point $\tilde{x}$ in the domain rather than at the discrete points for which we have data. Thus we write the approximate solution as:

$$\begin{aligned} u^h(x)|_{\tilde{x}} &= \sum_j^m b_j(x) c_j(\tilde{x}) \\ &= \mathbf{b}^T(x)\mathbf{c}(\tilde{x}) \end{aligned} \tag{16}$$

and write the *local* objective function at point $\tilde{x}$ as:

$$\begin{aligned}\mathcal{R}^{MLS}|_{\tilde{x}} &= \sum_i^N w_i(x_i - \tilde{x}) \left(u^h(x_i) - u_i\right)^2 \\ &= \sum_i^N w_i(x_i - \tilde{x}) \left(\sum_j^m b_j(x_i)c_j(\tilde{x}) - u_i\right)^2\end{aligned} \tag{17}$$

to be minimized with respect to $\mathbf{c}(\tilde{x})$. Also, in the MLS formulation, note that the weights are actually continuous functions of the distance from the point $\tilde{x}$. Typically, the weight function is designed to be maximum at 0 and drop off to zero at some distance (typically ± 2). The neighborhood over which is the weight function is non-zero is called the *compact support* or just the *support*.

The normal equations at point $\tilde{x}$ become:

$$\begin{aligned}\sum_i^N \left(b_k(x_i)w_i(x_i - \tilde{x}) \sum_j^m c_j(\tilde{x})b_j(x_i)\right) &= \sum_i^N b_k(x_i)w_i(x_i - \tilde{x})u_i \\ \mathbf{B}^T\mathbf{W}(\tilde{x})\mathbf{B}\mathbf{c}(\tilde{x}) &= \mathbf{B}^T\mathbf{W}(\tilde{x})\mathbf{u}\end{aligned} \tag{18}$$

which are solved in the usual way. An illustration of the Moving Least Squares fit is shown for our example where a quadratic function is fitted to the data around the point $x = 2.5$. The weight function filters out the far field points and uses data only from $x = 1.0$, $x = 2.0$, $x = 3.0$ and $x = 4.0$.

It is possible to reformulate the above equations using a local coordinate relative to the point $\tilde{x}$ around which the local least squares fit is being computed. Then the approximate solution is written as:

$$\begin{aligned}u^h(x)|_{\tilde{x}} &= \sum_j^m c_j(\tilde{x})b_j(x - \tilde{x}) \\ &= \mathbf{c}^T(\tilde{x})\mathbf{b}(x - \tilde{x})\end{aligned} \tag{19}$$

The basis vector $\mathbf{b}(x - \tilde{x})$ is called the *shifted basis* [1] or *centered basis* [2] and is denoted by $p(\tilde{x}, x)$. When x coincides with a particular data point x_i we use the shorthand notation $p_i(x)$ to denote $p(x, x_i)$. This reformulation with respect to a local coordinate system is said to typically lead to moment matrices that are better conditioned in the normal equations. However, instead of just presenting a modified residual with the shifted basis we present a more general technique involving the estimation of the function and its derivatives using a shifted basis in the following section.

6 Local Regression Estimation

Dilts [3] approached the problem of estimating a function (and its derivatives) using a method called Local Regression Estimation drawing upon concepts developed in statistics long ago. We attempt to present an outline of this powerful method here.

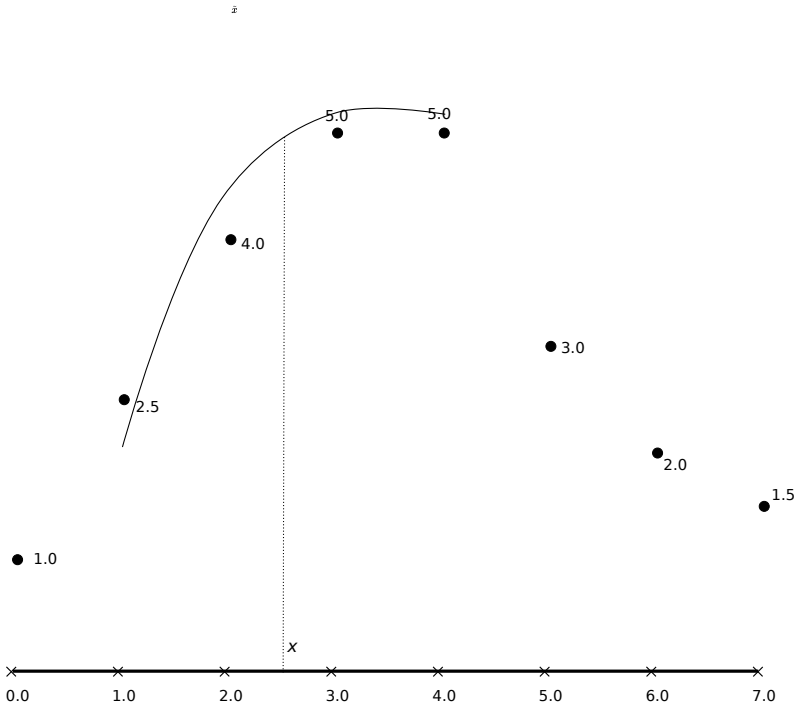

Figure 3: Illustration of a quadratic fit at an arbitrary point ($x = 2.5$) in the domain using Moving Least Squares.

The Taylor series expansion of an approximate solution u^h about point $\tilde{x}$ at a nearby point x_i is given by:

$$\begin{aligned} u^h(x_i) &= u^h(\tilde{x}) + \frac{\partial u^h}{\partial x}(x_i - \tilde{x}) + \frac{\partial^2 u^h}{\partial x^2}\frac{(x_i - \tilde{x})^2}{2!} + \frac{\partial^3 u^h}{\partial x^3}\frac{(x_i - \tilde{x})^3}{3!} + \dots \\ &= \boldsymbol{\beta}(\tilde{x})^T \mathbf{b}(x - \tilde{x}) \\ &= \boldsymbol{\beta}(\tilde{x})^T \mathbf{p}(\tilde{x}, x) \end{aligned} \tag{20}$$

where $\boldsymbol{\beta}(\tilde{x}) = \begin{bmatrix} u^h(\tilde{x}) & \frac{\partial u^h}{\partial x} & \frac{\partial^2 u^h}{\partial x^2} & \frac{\partial^3 u^h}{\partial x^3} & \dots \end{bmatrix}$.

Using this expansion, we can rewrite the usual least squares residual as:

$$\begin{aligned} \mathcal{R}^{LRE}|_{\tilde{x}} &= \sum_i^N w_i(x_i - \tilde{x}) \left(u^h(x_i) - u_i\right)^2 \\ &= \sum_i^N w_i(x_i - \tilde{x}) \left(\boldsymbol{\beta}(\tilde{x})\mathbf{p}_i(\tilde{x}) - u_i\right)^2 \end{aligned} \tag{21}$$

To find the extrema of this residual, we set its partial derivative with respect to each component β_k to zero to get:

$$\begin{aligned} \sum_i^N 2w_i(x_i - \tilde{x})p_{ik}(\tilde{x}) \left(\boldsymbol{\beta}(\tilde{x})\mathbf{p}_i(\tilde{x}) - u_i\right) &= 0 \\ \sum_i^N \boldsymbol{\beta}(\tilde{x})p_{ik}(\tilde{x})w_i(x_i - \tilde{x})\mathbf{p}_i(\tilde{x}) &= \sum_i^N w_i(x_i - \tilde{x})p_{ik}(\tilde{x})u_i \end{aligned} \tag{22}$$

Combining m such equations, we can write:

$$\mathbf{P}^T(\tilde{x})\mathbf{W}(\tilde{x})\mathbf{P}(\tilde{x})\boldsymbol{\beta}(\tilde{x}) = \mathbf{P}^T(\tilde{x})\mathbf{W}(\tilde{x})\mathbf{u} \tag{23}$$

Then we can solve for $\boldsymbol{\beta}(\tilde{x})$ as:

$$\boldsymbol{\beta}(\tilde{x}) = \boldsymbol{\psi}^T(\tilde{x})\mathbf{u} \tag{24}$$

where $\boldsymbol{\psi}(\tilde{x}) = \left(\mathbf{P}^T(\tilde{x})\mathbf{W}(\tilde{x})\mathbf{P}(\tilde{x})\right)^{-1}\mathbf{P}^T(\tilde{x})\mathbf{W}(\tilde{x})$.

Furthermore, Dilts introduces an $m \times m$ matrix called the *Taylor Series Jet Matrix* whose columns consist of the first m derivatives of $b(x)$, i.e.,

$$\mathbf{J}_b(x) = \begin{bmatrix} \mathbf{b}(x) & \mathbf{b}'(x) & \mathbf{b}''(x) & \mathbf{b}'''(x) \end{bmatrix} \tag{25}$$

and shows that for all bases composed of monomials of up to degree 3 in dimensions up to 4, the following holds:

$$\mathbf{J}_b^{-1}(x) = \mathbf{J}_b(-x) \tag{26}$$

leading to the equation:

$$\mathbf{p}(\tilde{x}, x) = \mathbf{b}(x - \tilde{x}) = \mathbf{J}_b^{-1}(\tilde{x})\mathbf{b}(x) \tag{27}$$

where x is another point in the domain.

Given the definition of $\boldsymbol{\beta}(x)$, we can conclude that:

$$\begin{aligned}
\boldsymbol{\beta}(x) &= \begin{bmatrix} u^h(x) & \frac{\partial u^h}{\partial x} & \frac{\partial^2 u^h}{\partial x^2} & \frac{\partial^3 u^h}{\partial x^3} & \dots \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{c}^T(x)\mathbf{b}(x) & \mathbf{c}^T(x)\mathbf{b}'(x) & \mathbf{c}^T(x)\mathbf{b}''(x) & \mathbf{c}^T(x)\mathbf{b}'''(x) & \dots \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{c}^T(x) \begin{bmatrix} \mathbf{b}(x) & \mathbf{b}'(x) & \mathbf{b}''(x) & \mathbf{b}'''(x) & \dots \end{bmatrix} \end{bmatrix} \\
&= \mathbf{c}^T(x)\mathbf{J}_b(x)
\end{aligned} \tag{28}$$

Therefore, knowing $\boldsymbol{\beta}(\tilde{x})$, we can compute $\mathbf{c}(\tilde{x})$ as:

$$\mathbf{c}(\tilde{x}) = (\mathbf{J}_b^T(\tilde{x}))^{-1}\boldsymbol{\beta}^T(\tilde{x}) \tag{29}$$

We can also derive the Local Regression Residual from the Moving Least Squares Residual (adapted from Dilts[3]) as shown below:

$$\begin{aligned}
\mathcal{R}^{MLS}|_{\tilde{x}} &= \sum_i^N w_i(x_i - \tilde{x}) \left(u^h(x_i)|_{\tilde{x}} - u_i \right)^2 \\
&= \sum_i^N w_i(x_i - \tilde{x}) \left(\mathbf{c}^T(\tilde{x})\mathbf{b}(x_i) - u_i \right)^2 \\
&= \sum_i^N w_i(x_i - \tilde{x}) \left(\mathbf{c}^T(\tilde{x})\mathbf{J}_b(\tilde{x})\mathbf{J}_b^{-1}(\tilde{x})\mathbf{b}(x_i) - u_i \right)^2 \\
&= \sum_i^N w_i(x_i - \tilde{x}) \left(\boldsymbol{\beta}(\tilde{x})\mathbf{p}(\tilde{x}, x_i) - u_i \right)^2 \\
&= \sum_i^N w_i(x_i - \tilde{x}) \left(\boldsymbol{\beta}(\tilde{x})\mathbf{p}_i(\tilde{x}) - u_i \right)^2 \\
&= \mathcal{R}^{LRE}
\end{aligned} \tag{30}$$

The distinguishing feature of Local Regression Estimation compared to Moving Least Squares is that by embedding the derivatives of approximation directly into the residual, Local Regression Estimation fits both the function and its derivatives to the data. This leads to smoother derivatives than other methods where only the function is fit to the data and the derivatives have to be computed by explicitly differentiating the

approximate function. Not only that, Local Regression Estimation readily computes the derivatives of the approximating function as a result of the estimation process as:

$$\begin{aligned} u^h(\tilde{x}) &= \boldsymbol{\beta}(\tilde{x})\mathbf{e}_0 = \mathbf{c}^T(\tilde{x})\mathbf{J}_b(\tilde{x})\mathbf{e}_0 \\ \frac{\partial u^h(\tilde{x})}{\partial \tilde{x}} &= \boldsymbol{\beta}(\tilde{x})\mathbf{e}_1 = \mathbf{c}^T(\tilde{x})\mathbf{J}_b(\tilde{x})\mathbf{e}_1 \\ \frac{\partial^2 u^h(\tilde{x})}{\partial \tilde{x}^2} &= \boldsymbol{\beta}(\tilde{x})\mathbf{e}_2 = \mathbf{c}^T(\tilde{x})\mathbf{J}_b(\tilde{x})\mathbf{e}_2 \\ &\vdots \end{aligned} \tag{31}$$

where $\mathbf{e}_i$ is the i^{th} Cartesian basis vector.

7 Acknowledgements

This work was performed under the auspices of the National Nuclear Security Administration of the US Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396 and supported by the DOE Advanced Simulation and Computing (ASC) program.

The author wishes to acknowledge Gary Dilts for the many discussions on Local Regression Estimation and his help in revising this document.

References

[1] Belytschko, T., Kronzaug, Y., Fleming, M., Organ, D. and Liu, W.K., "Smoothing and Accelerated Computations in the Element Free Galerkin Method." *Journal of Computational and Applied Mathematics*, 74:111-126, 1996.

[2] Breitkopf, P., Rassineux, A. and Villon, P. "An introduction to Moving Least Squares Meshfree Methods," *Revue Euopéenne des Éléments Finis*, 11:7-8, 825-867, DOI: 10.3166/reef.11.825-826, 2002.

[3] Dilts, G.A. "Estimation of Integral Operators on Random Data," Los Alamos Technical Report, LA-UR-17-23408, Los Alamos National Laboratory, Los Alamos, NM, 2017.