Luis Fernando Verduzco Martínez
Department of Computational Science - CIMAT
Interpolation with the Finite Element Method for surfaces.

Contents

1 Resume

It is presented the Finite Element formulation and computational implementation for the interpolation of 3D surfaces. The four-node bilinear elements are used for discretization, both parametric and natural-coordinated, although the mesh generated is uniform with squared elements. Three 3D functions are considered for experimentation and comparison of results regarding the performance of the method for various parameters (specially for the number of nodes, λ_x and λ_y). The method showed to have a great performance for only a few sample data points and number of finite elements which makes it quite advantageous when computational power is not available.

2 Introduction

In many engineering problems it is required to estimate variational functions in 3D dimensions. In topography surveys, for instance, such variational functions can represent the surface of a terrain that needs to be known in as much detail as possible for the estimation of earth-movement volumes or to make an architectural infrastructure project. For Structural Engineers such variational functions could represent the deformed surface of a shell, membrane or plate subject to certain stresses and forces.

Similar as for the interpolation of curves in 2D, the Finite Element Method can also used to approximate surfaces within a certain range space. The *Least Square* principle is used as well for the Finite Element formulation through the approximating shape functions, which may vary in grade according to which type of finite element may be chosen for discretization.

3 Discretization

Let us assume we have a set of data sample points as shown in **Fig. 1 (Left)** (these points could be thought of to be elevations over a terrain obtained from a topographical survey) from which it is required to approximate the shape of such terrain (see **Fig. 1 (Right)**).

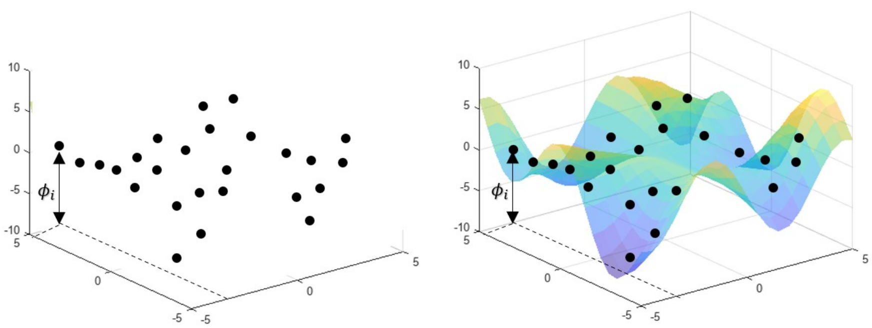

Figure 1: *(Left) Set of sample data points from which to approximate a surface, (Right) Approximating surface.*

In order to perform such approximation for the interpolating surface a linear system of equations (1) must be solved for the unknowns ϕ_i, where C_x and C_y are the regularization factor matrices for each axis direction that allow for the proper or wished adjustment of the surface to minimize the error degree, K is the shape functions matrix that depends entirely on the sample data points and F is the force vector representing weight of the sample data points on each finite element to which the solution must be approximated.

$$([K] + [C_x] + [C_y])[\phi] = [F] \tag{1}$$

For the 2D case the size of each matrix in the previous equation for each finite element was $2x2$ given that each finite element was a line of two nodes. However, for the 3D case the size of such matrices will depend on the finite element shape. In the present work the bilinear finite element (see **Fig. 2**) is considered, consisting of four nodes, therefore, the size of each matrix of each finite element will be $4x4$.

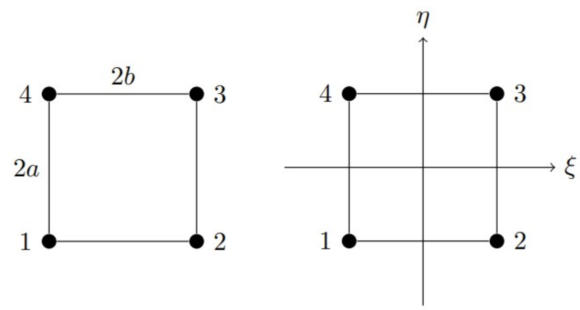

Figure 2: *(Left) Natural coordinate bilinear element, (Right) Parametric bilinear element.*

4 Interpolating Shape functions

The shape functions of the bilinear element in *natural coordinates* (**Fig. 2** (Left)) are defined as (2), and for parametric coordinates as (3). Parametric coordinated shape functions are preferred to work with as they allow the bilinear finite elements to have curved sides which would adapt better to complex and irregular surfaces. For the present work, the matrix K of equation (1) will be formulated in natural coordinates and C_x and C_y in parametric coordinates, given that it is required for the latter matrices to be integrated over the area of their respective finite element domain. For such case, the Gaussian quadrature can be deployed with by using the weight functions and gauss points presented in **Table 1** (which for the present problem, only two gauss points $n = 2$ would apply):

$$\begin{aligned} N_1 &= \tfrac{1}{4ab}(x - x_2)(y - y_4) \\ N_2 &= \tfrac{1}{4ab}(x - x_1)(y - y_3) \\ N_3 &= \tfrac{1}{4ab}(x - x_4)(y - y_2) \\ N_4 &= \tfrac{1}{4ab}(x - x_3)(y - y_1) \end{aligned} \tag{2}$$

$$\begin{aligned} \bar{N}_1 &= \tfrac{1}{4}(\xi - 1)(\eta - 1) \\ \bar{N}_2 &= -\tfrac{1}{4}(\xi + 1)(\eta - 1) \\ \bar{N}_3 &= \tfrac{1}{4}(\xi + 1)(\eta - 1) \\ \bar{N}_4 &= -\tfrac{1}{4}(\xi - 1)(\eta + 1) \end{aligned} \tag{3}$$

Similar as for the case in 2D, the quality of the results will depend directly on these shape functions. There are other bilinear elements that consist of more nodes (until 20 or more nodes) from which better interpolating results can be obtained, although with more computational power. Such *shape functions* guarantee the necessary compatibility between each ϕ_i of each finite element's node in their intersection between other neighbour finite element's nodes. In **Fig. 3** it is shown the interpolation shape of N_1 and N_2 for a four-node bilinear element:

Table 1: *Gauss points and weight functions to apply Gaussian quadrature.*

n	ξ_i	W_i
1	0	2
2	$\pm\frac{1}{\sqrt{3}}$	1

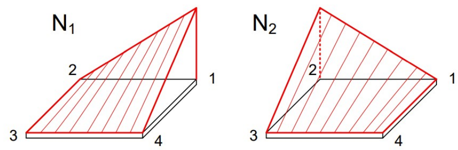

Figure 3: *Interpolation functions for a 4-node bilinear element.*

4.1 Coordinate interpolation

The previous Shape function can then be used to approximate the function values globally on each node of each finite element from the local coordinates of each element as (4), where n is the number of nodes:

$$\begin{aligned} x &= \textstyle\sum_1^n N_i(\xi,\eta)x_i \\ y &= \textstyle\sum_1^n N_i(\xi,\eta)y_i \end{aligned} \tag{4}$$

From which the following partial derivatives can be obtained (5):

$$\begin{aligned} \frac{\partial N_i}{\partial \xi} &= \frac{\partial N_i}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial N_i}{\partial y}\frac{\partial y}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} &= \frac{\partial N_i}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial N_i}{\partial y}\frac{\partial y}{\partial \eta} \end{aligned} \tag{5}$$

and expressed in matrix form as (6):

$$\begin{bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \end{bmatrix} \tag{6}$$

4.2 Functional interpolation

The same procedure can be applied to interpolate the function values ϕ as (7):

$$\phi_i(x,y) = \textstyle\sum N_i(\xi,\eta)\phi_i \tag{7}$$

from which the partial derivatives (8) can be obtained:

$$\begin{aligned} \frac{\partial \phi}{\partial x} &= \textstyle\sum \frac{\partial N_i}{\partial x}\phi_i \\ \frac{\partial \phi}{\partial y} &= \textstyle\sum \frac{\partial N_i}{\partial y}\phi_i \end{aligned} \tag{8}$$

and expressed in matrix form as (9) and (10), respectively:

$$\phi(x,y) = \begin{bmatrix} N_1 & N_2 & N_3 & N_4 \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} \nabla\phi_x \\ \nabla\phi_y \end{bmatrix} = \begin{bmatrix} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial x} & \frac{\partial N_4}{\partial x} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_2}{\partial y} & \frac{\partial N_3}{\partial y} & \frac{\partial N_4}{\partial y} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} \tag{10}$$

5 FEM formulation by Least-Square regression

An error function E sought to be minimized for each interpolated node can be expressed as in a Least-Square form (11):

$$E = \frac{1}{2}\sum_k (\phi(x_k, y_k) - Z_k)^2 + \frac{\lambda_x}{2}\int_\Omega (\nabla\phi_x)^2 d\Omega + \frac{\lambda_y}{2}\int_\Omega (\nabla\phi_y)^2 d\Omega \tag{11}$$

Thus, by computing its derivative and make it equal to zero, it yields (12):

$$\begin{aligned} \frac{\partial E}{\partial \phi_i} = \sum_k (\phi(x_k, y_k) - Z_k)N + \lambda_x \int_\Omega (\frac{\partial}{\partial\phi})[\frac{\partial N}{\partial x}(\phi^T)\frac{\partial N^T}{\partial x}(\phi)]d\Omega + ... \\ \lambda_y \int_\Omega (\frac{\partial}{\partial\phi})[\frac{\partial N}{\partial y}(\phi^T)\frac{\partial N^T}{\partial y}(\phi)]d\Omega = 0 \end{aligned} \tag{12}$$

Now, by re-arranging terms, the previous equations can be expressed as (13):

$$\begin{aligned} \sum_k \phi(x_k, y_k)N + \lambda_x \int_\Omega (\frac{\partial}{\partial\phi})[\frac{\partial N}{\partial x}(\phi^T)\frac{\partial N^T}{\partial x}(\phi)]d\Omega + ... \\ \lambda_y \int_\Omega (\frac{\partial}{\partial\phi})[\frac{\partial N}{\partial y}(\phi^T)\frac{\partial N^T}{\partial y}(\phi)]d\Omega = \sum_k Z_k N \end{aligned} \tag{13}$$

And as a linear system of equations (14), where k is the number of sample points within each finite element:

$$(\sum_k \begin{bmatrix} N_1N_1 & N_1N_2 & N_1N_3 & N_1N_4 \\ N_2N_1 & N_2N_2 & N_2N_3 & N_2N_4 \\ N_3N_1 & N_3N_2 & N_3N_3 & N_3N_4 \\ N_4N_1 & N_4N_2 & N_4N_3 & N_4N_4 \end{bmatrix} + \lambda_x \int_\Omega \begin{bmatrix} \frac{\partial N_1}{\partial x}\frac{\partial N_1}{\partial x} & \frac{\partial N_1}{\partial x}\frac{\partial N_2}{\partial x} & \frac{\partial N_1}{\partial x}\frac{\partial N_3}{\partial x} & \frac{\partial N_1}{\partial x}\frac{\partial N_4}{\partial x} \\ \frac{\partial N_2}{\partial x}\frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x}\frac{\partial N_2}{\partial x} & \frac{\partial N_2}{\partial x}\frac{\partial N_3}{\partial x} & \frac{\partial N_2}{\partial x}\frac{\partial N_4}{\partial x} \\ \frac{\partial N_3}{\partial x}\frac{\partial N_1}{\partial x} & \frac{\partial N_3}{\partial x}\frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial x}\frac{\partial N_3}{\partial x} & \frac{\partial N_3}{\partial x}\frac{\partial N_4}{\partial x} \\ \frac{\partial N_4}{\partial x}\frac{\partial N_1}{\partial x} & \frac{\partial N_4}{\partial x}\frac{\partial N_2}{\partial x} & \frac{\partial N_4}{\partial x}\frac{\partial N_3}{\partial x} & \frac{\partial N_4}{\partial x}\frac{\partial N_4}{\partial x} \end{bmatrix} + \ldots$$

$$\ldots + \lambda_y \int_\Omega \begin{bmatrix} \frac{\partial N_1}{\partial y}\frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial y}\frac{\partial N_2}{\partial y} & \frac{\partial N_1}{\partial y}\frac{\partial N_3}{\partial y} & \frac{\partial N_1}{\partial x}\frac{\partial N_4}{\partial y} \\ \frac{\partial N_2}{\partial y}\frac{\partial N_1}{\partial y} & \frac{\partial N_2}{\partial y}\frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial y}\frac{\partial N_3}{\partial y} & \frac{\partial N_2}{\partial y}\frac{\partial N_4}{\partial y} \\ \frac{\partial N_3}{\partial x}\frac{\partial N_1}{\partial y} & \frac{\partial N_3}{\partial y}\frac{\partial N_2}{\partial y} & \frac{\partial N_3}{\partial y}\frac{\partial N_3}{\partial y} & \frac{\partial N_3}{\partial y}\frac{\partial N_4}{\partial y} \\ \frac{\partial N_4}{\partial y}\frac{\partial N_1}{\partial y} & \frac{\partial N_4}{\partial y}\frac{\partial N_2}{\partial y} & \frac{\partial N_4}{\partial y}\frac{\partial N_3}{\partial y} & \frac{\partial N_4}{\partial y}\frac{\partial N_4}{\partial y} \end{bmatrix}) \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} = \sum_k \begin{bmatrix} Z_kN_1 \\ Z_kN_2 \\ Z_kN_3 \\ Z_kN_4 \end{bmatrix} \tag{14}$$

Finally, when integrating the regularization factor matrices with the Gaussian quadrature we obtain (15):

$$\begin{aligned} &(\sum_k \begin{bmatrix} N_1N_1 & N_1N_2 & N_1N_3 & N_1N_4 \\ N_2N_1 & N_2N_2 & N_2N_3 & N_2N_4 \\ N_3N_1 & N_3N_2 & N_3N_3 & N_3N_4 \\ N_4N_1 & N_4N_2 & N_4N_3 & N_4N_4 \end{bmatrix} + \ldots \\ &\lambda_x \int_{-1}^{1}\int_{-1}^{1} \begin{bmatrix} \frac{\partial N_1}{\partial x}\frac{\partial N_1}{\partial x} & \frac{\partial N_1}{\partial x}\frac{\partial N_2}{\partial x} & \frac{\partial N_1}{\partial x}\frac{\partial N_3}{\partial x} & \frac{\partial N_1}{\partial x}\frac{\partial N_4}{\partial x} \\ \frac{\partial N_2}{\partial x}\frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x}\frac{\partial N_2}{\partial x} & \frac{\partial N_2}{\partial x}\frac{\partial N_3}{\partial x} & \frac{\partial N_2}{\partial x}\frac{\partial N_4}{\partial x} \\ \frac{\partial N_3}{\partial x}\frac{\partial N_1}{\partial x} & \frac{\partial N_3}{\partial x}\frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial x}\frac{\partial N_3}{\partial x} & \frac{\partial N_3}{\partial x}\frac{\partial N_4}{\partial x} \\ \frac{\partial N_4}{\partial x}\frac{\partial N_1}{\partial x} & \frac{\partial N_4}{\partial x}\frac{\partial N_2}{\partial x} & \frac{\partial N_4}{\partial x}\frac{\partial N_3}{\partial x} & \frac{\partial N_4}{\partial x}\frac{\partial N_4}{\partial x} \end{bmatrix} detJd\xi d\eta + \ldots \\ &\ldots + \lambda_y \int_{-1}^{1}\int_{-1}^{1} \begin{bmatrix} \frac{\partial N_1}{\partial y}\frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial y}\frac{\partial N_2}{\partial y} & \frac{\partial N_1}{\partial y}\frac{\partial N_3}{\partial y} & \frac{\partial N_1}{\partial x}\frac{\partial N_4}{\partial y} \\ \frac{\partial N_2}{\partial y}\frac{\partial N_1}{\partial y} & \frac{\partial N_2}{\partial y}\frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial y}\frac{\partial N_3}{\partial y} & \frac{\partial N_2}{\partial y}\frac{\partial N_4}{\partial y} \\ \frac{\partial N_3}{\partial x}\frac{\partial N_1}{\partial y} & \frac{\partial N_3}{\partial y}\frac{\partial N_2}{\partial y} & \frac{\partial N_3}{\partial y}\frac{\partial N_3}{\partial y} & \frac{\partial N_3}{\partial y}\frac{\partial N_4}{\partial y} \\ \frac{\partial N_4}{\partial y}\frac{\partial N_1}{\partial y} & \frac{\partial N_4}{\partial y}\frac{\partial N_2}{\partial y} & \frac{\partial N_4}{\partial y}\frac{\partial N_3}{\partial y} & \frac{\partial N_4}{\partial y}\frac{\partial N_4}{\partial y} \end{bmatrix} detJd\xi d\eta) \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} = \sum_k \begin{bmatrix} Z_kN_1 \\ Z_kN_2 \\ Z_kN_3 \\ Z_kN_4 \end{bmatrix} \end{aligned} \tag{15}$$

6 Results

The following three 3D functions (16), (17) and (18), are considered to shown-case the performance of this method:

6.1 Function 1

$$f(x,y) = cos(x)cos(y)exp(\sqrt{\frac{x^2+y^2}{5}}) \tag{16}$$

$$\lambda_x = \lambda_y = 500, n - nodes = 20^2$$

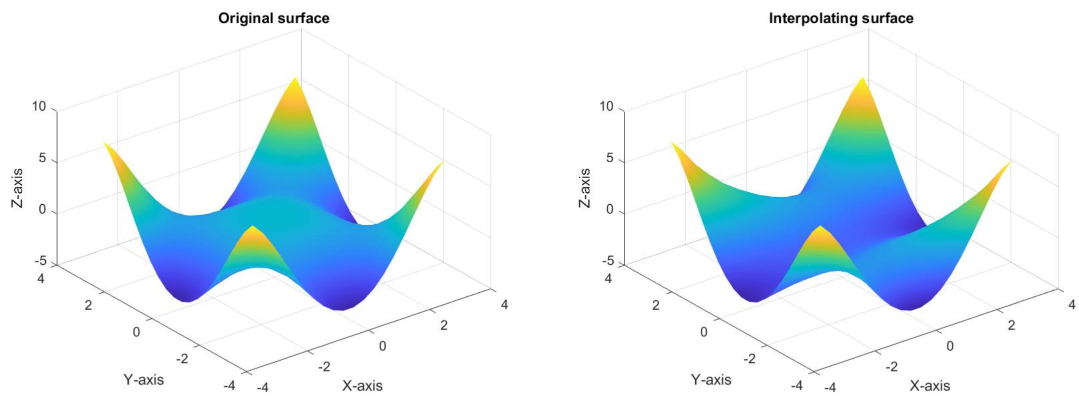

Figure 4: *(Left) Visualization of function 1, (Right) Visualization of the interpolating surface of function 1.*

6.2 Function 2

$$f(x, y) = cos(x)y \tag{17}$$

$$\lambda_x = \lambda_y = 500, n - nodes = 20^2$$

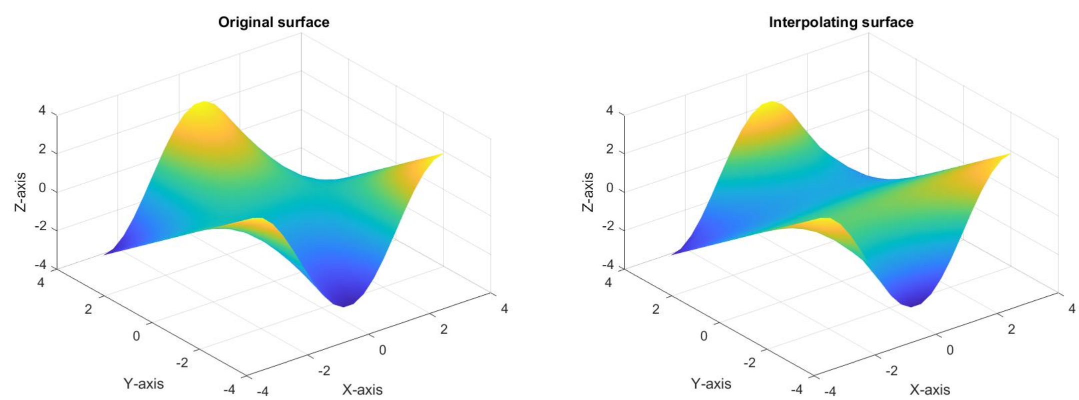

Figure 5: *(Left) Visualization of function 2, (Right) Visualization of the interpolating surface of function 2.*

6.3 Function 3

$$f(x,y) = ((x.^2 + y - 11)^2 + (x + y.^2 - 7).^2); \tag{18}$$

$$\lambda_x = \lambda_y = 0.85, n - nodes = 40^2$$

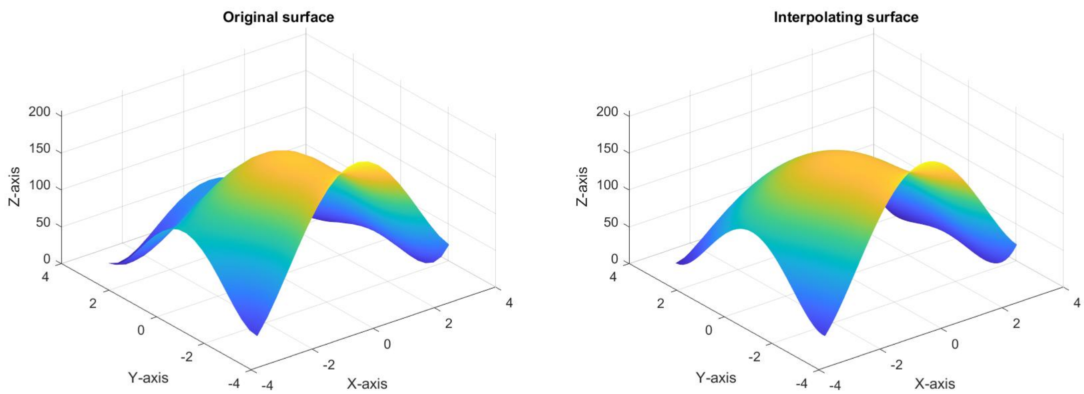

Figure 6: *(Left) Visualization of function 3, (Right) Visualization of the interpolating surface of function 3.*

References

[1] J.N. Reddy, An Introduction to the Finite Element Method, 2nd ed. McGraw-Hill, 1993

[2] R.L. Burden, D.J. Faires, A.M. Burden, Numerical Analysis, 10th ed. Cengage Learning, 2014