Hindawi
Journal of Applied Mathematics
Volume 2022, Article ID 8227086, 8 pages
https://doi.org/10.1155/2022/8227086

Research Article

Lagrange Multivariate Polynomial Interpolation: A Random Algorithmic Approach

A. Essanhaji and M. Errachid

Laboratory of Mathematics, Computing and Applications-Information Security (LabMiA-SI), Mohammed V University in Rabat, Centre Régional des Métiers de l'Enseignement et de la Formation (CRMEF) de Rabat, 1 Avenue Allal Alfassi, Madinat Al Irfane, B.P. 6210, 10 000 Rabat, Morocco

Correspondence should be addressed to A. Essanhaji; abdelhakessanhaji@gmail.com

Received 5 January 2022; Revised 27 January 2022; Accepted 14 February 2022; Published 14 March 2022

Academic Editor: Saeid Abbasbandy

The problems of polynomial interpolation with several variables present more difficulties than those of one-dimensional interpolation. The first problem is to study the regularity of the interpolation schemes. In fact, it is well-known that, in contrast to the univariate case, there is no universal space of polynomials which admits unique Lagrange interpolation for all point sets of a given cardinality, and so the interpolation space will depend on the set Z of interpolation points. Techniques of univariate Newton interpolating polynomials are extended to multivariate data points by different generalizations and practical algorithms. The Newton basis format, with divided-difference algorithm for coefficients, generalizes in a straightforward way when interpolating at nodes on a grid within certain schemes. In this work, we propose a random algorithm for computing several interpolating multivariate Lagrange polynomials, called RLMVPIA (Random Lagrange Multivariate Polynomial Interpolation Algorithm), for any finite interpolation set. We will use a Newton-type polynomials basis, and we will introduce a new concept called (Z, z)-partition. All the given algorithms are tested on examples. RLMVPIA is easy to implement and requires no storage.

1. Introduction

Let $\mathbb{K}$ be a commutative field and $p, n \in \mathbb{N}^*$. By $\Pi^p = \mathbb{K}[x_1, \cdots, x_p]$, we denote the algebra of all polynomials in p variables, and we denote by Π_d^p the subspace of all polynomials of total degree less than or equal to d, where d is a nonnull integer.

Given a finite interpolation set $Z_n = \{z_1, \cdots z_n\} \subset \mathbb{K}^p$ of distincts nodes, the Lagrange interpolation problem consists of finding, for a given data vector $R = (r_z : z \in Z_n) \in \mathbb{K}^{Z_n}$, a polynomial $P \in \Pi^p$ such that

$$P(Z_n) = R, \text{ that is, } P(z) = r_z, z \in Z_n. \tag{1}$$

We will then say that P is an interpolating polynomial for R on Z. More precisely, Z_n is called poised or correct or unisolvent [1–3] for a subspace $\mathscr{P}$ of Π^p; if the Lagrange interpolation problem (1) has a unique interpolating polynomial in $\mathscr{P}$ for any given data vector $R = (r_z : z \in Z_n) \in \mathbb{K}^{Z_n}$, it means, in other words, that the function

$$P \in \mathscr{P} \mapsto (P(z) : z \in Z_n) \in \mathbb{K}^{Z_n} \tag{2}$$

is a linear isomorphism. Then, it is necessary that $\dim \mathscr{P} = n$. The problem of researching such subspaces will be denoted $\mathscr{P}(Z_n)$.

In this article, we construct a random algorithm for finding several sub-spaces solutions of the problem $\mathscr{P}(Z_n)$.

It is well-known that in the univariate case ($p = 1$) the Lagrange interpolation problem with respect to n distinct points is always uniquely solvable, if one takes $\mathscr{P}$ to be the space of all polynomials of degree less than or equal to $n - 1$. In several variables, however, the situation is much more

difficult. In order to successfully interpolate Z_n on Π_d^p, we must have

$$n = \binom{p+d}{p}, \tag{3}$$

since

$$\dim \Pi_d^p = \binom{p+d}{p}. \tag{4}$$

And even if this is the case, there can be the problem that the points lie on some algebraic surface of degree d; i.e., there is some polynomial Q of total degree at most d which vanishes on Z_n. For example, take $p = 2$, $d = 1$, and $Z = \{(-1, 0), (0, 0), (1, 0)\}$; it is easy to see that the set Z is not poised on the space $\Pi_1^2 = \text{span}\{1, x, y\}$ (since $P = y$ vanishes on Z).

So the poisedness of multivariate polynomial interpolation depends on the geometric structure of the interpolation set Z_n. Tensor product interpolation is the oldest particular case extending the univariate theory where the interpolation set and space are obtained by tensor products of the univariate ones. The Lagrange formula and the Newton formula with divided differences are easily extended to this problem, as can be found in [4–15].

In a recent publication [7], the author proposed a generalization of the univariate program of Newton form basis and divided-difference algorithm in $\mathbb{K}^p$. The required interpolation sets are those which admit an indexation having a regular structure as triangular, rectangular, or more generally a lower set [2, 7, 8, 16–18]. He shows how the index set structure appropriately determines the interpolation space. When the sum of indices is bounded by d, there is a unique interpolation with a polynomial of degree $\leq d$.

In [5], by using the Schur complements and the Sylvester identity, the authors established the RMVPIA (Recursive MultiVariate Polynomial Interpolation Algorithm) when the interpolation set is a full grid.

Polynomial interpolation with several variables occurs in several topics of applied mathematics and engineering [19–22], hence the interest in seeking consistent and simple to implement polynomial interpolation algorithms.

In this work, we propose a new algorithm for computing several interpolation spaces for any finite interpolation set. This algorithm which is called RLMVPIA (Random Lagrange MultiVariate Polynomial Interpolation Algorithm) is based on a recursive random scheme. RLMVPIA allows us to simultaneously determine interpolating polynomials. For that, we will use a Newton-type polynomials basis, and we will introduce a new concept called (Z, z)-partition. All the given algorithms are tested on examples. As RMVPIA, RLMVPIA is easy to implement and requires no storage.

The principle of our approach is to solve $\mathscr{P}(Z_n)$ knowing a solution of $\mathscr{P}(Z_{n-1})$, where $Z_{n-1} = \{z_1, \cdots, z_{n-1}\}$ is a subset of Z_n with $n-1$ nodes. More precisely, if $\mathscr{P}_{n-1}$ is an interpolation space for Z_{n-1}, and $P_{n-1} \in \mathscr{P}_{n-1}$ verifying $P_{n-1}(z) = r_z$, $z \in Z_{n-1}$; then, we construct, in a way, a polynomial Q_n verifying

$$\begin{gathered} Q_n(z) = 0, \forall z \in Z_{n-1}, \\ Q_n(z_n) \neq 0. \end{gathered} \tag{5}$$

So $\mathscr{P}_n = \mathscr{P}_{n-1} + \mathbb{K}Q_n$ is a solution of $\mathscr{P}(Z_n)$, and the solution of (1) in $\mathscr{P}_n$ is a polynomial in the form

$$P_n = P_{n-1} + s_n Q_n, \tag{6}$$

where s_n is a scalar to compute.

This work is organized as follows: in Section 2 we present the notion of (Z, z)-partition. In Section 3, we give the algorithm RLMVPIA. In Section 4, we illustrate our algorithms by different examples.

2. (Z, z)-Partition Concept

This new approach takes into consideration the distribution of the nodes of the considered interpolation set Z_n by introducing a new concept described in the following. We define a notion of (Z, z)-partition, and we present a random algorithm for computing the polynomials Q_i, for $i \in [[1 ; n]]$, which will be used for giving the algorithm RLMVPIA.

In this section, Z is a finite set of $\mathbb{K}^p$, $p \in \mathbb{N}^*$, and $z \in \mathbb{K}^p \setminus Z$. For $k \in \{1, \cdots, p\}$, we note x_k the canonical coordinated form, defined as

$$\begin{array}{rccc} x_k : & \mathbb{K}^p & \longrightarrow & \mathbb{K} \\ & t = (\alpha_1, \cdots, \alpha_p) & \mapsto & x_k(t) = \alpha_k \end{array}, \tag{7}$$

and $x_k(Z) = \{x_k(t) : t \in Z\}$.

Definition 1. For $k \in [[1 ; p]]$, let I_k be a subset of $x_k(Z)$.

(1) It will be said that $(I_1, \cdots, I_p)$ is a Z-partition if

$$\forall t \in Z, \exists k \in [[1 ; p]] : x_k(t) \in I_k \tag{8}$$

In this case, $\sum_{k=1}^{p} \text{card}\,(I_k)$ is called the length of the Z-partition $(I_1, \cdots, I_p)$.

(2) Let $z \in \mathbb{K}^p \setminus Z$. It is said that $(I_1, \cdots, I_p)$ is a (Z, z)-partition if

(a) $(I_1, \cdots, I_p)$ is a Z-partition

(b) $\forall k \in [[1 ; p]], x_k(z) \notin I_k$

Proposition 2. *For all $z \in \mathbb{K}^p \setminus Z$, a (Z, z)-partition still exists.*

Proof. To prove that, we show how to construct a (Z, z) -partition. Let ϕ be the function

$$\begin{aligned} Z &\longrightarrow [[1\,;p]], \\ t &\longrightarrow \max\{k : x_k(t) \neq x_k(z)\}. \end{aligned} \tag{9}$$

ϕ is well defined because $z \notin Z$. We set

$$\phi(Z) = \{p_1, \cdots, p_i\} \subset [[1\,;p]], \tag{10}$$

and for $k \in [[1\,;p]]$, we take

$$I_k = \begin{cases} x_k\left(\phi^{-1}(\{k\})\right), & \text{if } k \in \{p_1, \cdots, p_i\}, \\ \varnothing, & \text{else.} \end{cases} \tag{11}$$

Then $(I_1, \cdots, I_p)$ is a (Z, z)-partition.

To illustrate the proof of the proposition, we give below some examples in the cases $p = 2$ and $p = 3$ □

2.1. Case $p = 2$

Example 1. We set

$$Z = \{(1,1), (2,1), (0,2), (1,2), (2,2)\} \text{ and } z = (3,1). \tag{12}$$

We have $\phi(Z) = \{1, 2\}$ and

$$\phi^{-1}(\{1\}) = \{(1,1), (2,1)\}, \phi^{-1}(\{2\}) = \{(0,2), (1,2), (2,2)\}. \tag{13}$$

It follows that

$$I_1 = x_1\left(\phi^{-1}\right)(\{1\}) = \{1,2\}, I_2 = x_2\left(\phi^{-1}\right)(\{2\}) = \{2\}. \tag{14}$$

Example 2. We take

$$Z = \{(1,-1), (1,0), (1,1), (1,2), (1,3), (1,4), (5,4), (6,4), (7,4), (8,4), (9,4)\}, \tag{15}$$

with $z = (11, 5)$, and we have $\phi(Z) = \{2\}$, so we obtain

$$I_1 = \varnothing, I_2 = \{-1,\ 0,\ 1,\ 2,\ 3,\ 4\}. \tag{16}$$

2.2. Case $p = 3$

Example 3. We take

$$Z = \{(0,1,0), (1,1,0), (7,-1,0), (2,2,-1), (2,2,3), (0,3,1), (5,3,1), (-10,3,1)\}, \tag{17}$$

```
input: Z,  z
Initialization: I_1, ..., I_p : p empty sets
for t in Z (a random choice)
    Index = {1, ..., p}
    i = a random number of Index.
    while x_i(t) = x_i(z):
        remove i from Index
        i = a random number of Index
    endwhile
    remove i from Index
    for j in Index:
        if x_j(t) in I_j:
            pass to next element in Z
        endif
    endfor
    if x_i(t) not in I_i
        add x_i(t) to I_i
        Q = Q * (x_i - x_i(t))
    endif
endfor
output Q, I_1, ..., I_p
```

Algorithm 1: ZPNA

with $z = (2, 3, 1)$, and we have $\phi(Z) = \{1, 3\}$,

$$\begin{aligned} \phi^{-1}(\{1\}) &= \{(0,3,1), (5,3,1), (-10,3,1)\}, \\ \phi^{-1}(\{3\}) &= \{(0,1,0), (1,1,0), (7,-1,0), (2,2,-1), (2,2,3)\}, \end{aligned} \tag{18}$$

so

$$I_1 = \{0, 5,\ -10\},\ I_2 = \varnothing \text{ and } I_3 = \{0,\ -1,\ 3\}. \tag{19}$$

Now, one can easily get the following result.

Proposition 6. *Let $\mathcal{J} = (I_1, \cdots, I_p)$ be a (Z, z)-partition, and let $Q_{\mathcal{J}}$ be the polynomial associated, given by*

$$Q_{\mathcal{J}} = \prod_{k=1}^{p} \prod_{\alpha \in I_k} (x_k - \alpha), \tag{20}$$

with the convention that the product is equal to 1 when the set of indices is empty. So we have

$$\begin{aligned} &\forall t \in Z, \\ &Q_{\mathcal{J}}(t) = 0. \\ &Q_{\mathcal{J}}(z) \neq 0. \end{aligned} \tag{21}$$

The following algorithm called ZPNA ((Z, z)-Partition Newton Algorithm) randomly constructs a (Z, z)-partition and the associated polynomial.

Remark 7.

(1) All operations in the algorithm ZPNA have constant costs, so the complexity depends linearly on the length of Z

(2) For a (Z, z)-partition, the total degree of the associated polynomial is equal to the length of the partition which is less than $\sum_{k=1}^{p} \text{card}\,(x_k(Z))$

(3) The algorithm ZPNA is not deterministic. If we apply the algorithm several times, one can obtain several (Z, z)-partitions from which we can choose those of minimum length

Point 3 of the precedent remark can be illustrated by the following example:

$$Z = \{(1,-1), (1,0), (1,1), (1,2), (1,3), (1,4), (5,4), (6,4), (7,4), (8,4), (9,4)\}, \tag{22}$$

and $z = (11, 5)$. Applying the algorithm ZPNA several times, we get the following (Z, z)-partitions and the associated polynomials.

(1) $I_1 = \{1\}, I_2 = \{4\}$ and $Q = (x-1)(y-4)$

(2) $I_1 = \{1, 5, 6, 7\}, I_2 = \{-1, 4\}$ and $Q = (x-7)(x-6)(x-5)(x-1)(y-4)(y+1)$

(3) $I_1 = \varnothing$, $I_2 = \{-1, 0, 1, 2, 3, 4\}$ and $Q = y(y-4)(y+1)(y-1)(y-2)(y-3)$

(4) $I_1 = \{1, 5, 6, 7, 8\}, I_2 = \{-1, 0, 1, 2, 4\}$ and $Q = y(x-8)(x-7)(x-6)(x-5)(x-1)(y-4)(y-2)(y-1)(y+1)$

It is clear that the first solution is the best.

Theorem 8. *The algorithm ZPNA is correct.*

Proof. For proving that, we use loop invariant to help us understand why an algorithm is correct. We must show three things about a loop invariant:

(i) Initialization: it is true prior to the first iteration of the loop.

(ii) Maintenance: if it is true before an iteration of the loop, it remains true before the next iteration.

(iii) Termination: when the loop ends, the invariant gives us a useful property showing that the algorithm is correct.

For the ZPNA algorithm, we note for $k \in [[1\,;n]]$, $z_1, \cdots z_k$ the elements treated in the k first iterations, $Z_k = \{z_1, \cdots z_k\}$ et $I_1^k, \cdots, I_p^k$ the values of $I_1, \cdots, I_p$ in the iteration k. So the following property is a loop invariant: At the start of each iteration of the for loop, the $I_1^{k-1}, \cdots, I_p^{k-1}$ is a (Z_{k-1}, z) -partition.

For initialization, we start by showing that the loop invariant holds before the first loop iteration: when $k = 1$, the I_1^{k-1}, $\cdots$, I_p^{k-1} are empty, and no item is treated, we take $Z_0 = \varnothing$, so by convention, the $I_1^{k-1}, \cdots, I_p^{k-1}$ is a (Z_{k-1}, z)-partition.

For maintenance, next, we tackle the second property, showing that each iteration maintains the loop invariant. Assume that the $I_1^{k-1}, \cdots, I_p^{k-1}$ is a (Z_{k-1}, z)-partition, we note z_k the element chosen at the start of the iteration k, as $z \notin Z$, the loop (while $x_i(t) = x_i(z)$) ends, so we note i the index founded such as $x_i(z_k) \neq x_i(z)$. The analysis of the sequence of the iteration k makes it possible without difficulty to assert that $I_j^k = I_j^{k-1}$ if $j \neq i$ and for $j = i$ two cases arise: either $I_i^k = I_i^{k-1}$ if $x_j(z_k)$ is already present in one of the I_j, otherwise $I_i^k = I_i^{k-1} \cup \{x_i(z_k)\}$. In both cases, we have $I_1^k, \cdots, I_p^k$ is a (Z_k, z)-partition: for $t \in Z_k$, if $t \in Z_{k-1}$, then there is a $j \in [[1\,;n]]$ such as $x_j(t) \in I_j$ because $I_1^{k-1}, \cdots, I_p^{k-1}$ is a (Z_{k-1}, z)-partition; if $t = z_k$ by construction, we also have the result. On the other hand, as $x_j(z)$ is not in any of the $I_j^{k-1}, j \in [[1\,;n]]$, and as by construction $x_i(z_k) \neq x_i(z)$, it follows that $x_i(z) \notin I_i^k$, hence the result.

For termination, we examine finally what will happen when the loop terminates. When the loop terminates (for t in Z (a random choice)), the set $Z_n = Z$, with the invariant of the loop, we have $I_1^n, \cdots, I_p^n$ (which are the sets returned by the algorithm) which is a $(Z_n = Z, z)$-partition. Therefore, the algorithm is correct. □

3. RLMVPIA Random Approach

For solving, recursively, the problem $\mathscr{P}(Z_n)$, we start by choosing $\mathscr{P}_1 = \text{span}\{(1)\}$ as an obvious solution of the problem $\mathscr{P}(Z_1)$, since we have, for all $r_1 \in \mathbb{K}$, the constant polynomial

$$P_1 = r_1, \tag{23}$$

which is the interpolating polynomial of $R_1 = (r_1)$ on $Z_1 = \{z_1\}$ in $\mathscr{P}_1$. So we take $Q_1 = 1$ as a basis of $\mathscr{P}_1$ and we will use the notion of (Z, z)-partition for computing the polynomials Q_i, for $i \in [[2\,;n]]$, in order to give the algorithm RLMVPIA.

The following result shows how the solution of the problem $\mathscr{P}(Z_n)$ can be constructed recursively using the relationship (6).

Theorem 9. *For $k \in \{2, \cdots, n\}$, let $\mathscr{I}^{(k)} = (I_1^{(k)}, \cdots, I_p^{(k)})$ be a (Z_{k-1}, z_k)-partition, and let*

$$Q_k = \prod_{i=1}^{p} \prod_{\alpha \in I_i^{(k)}} (x_i - \alpha) \tag{24}$$

be the associated polynomial. Then, the space $\mathscr{P}_k = span\{Q_1, \cdots Q_k\}$ is a solution of the problem $\mathscr{P}(Z_k)$. More precisely, giving $R_k = (r_i : i = 1, \cdots, k)$ a data vector, the interpolating

polynomial for R_k on Z_k in $\mathscr{P}_k$ is given by P_k:

$$P_k = P_{k-1} + s_k Q_k, \tag{25}$$

where P_{k-1} is the interpolating polynomial for $R_{k-1} = (r_i : i = 1, \cdots, k-1)$ on Z_{k-1} in $\mathscr{P}_{k-1} = span\{Q_1, \cdots, Q_{k-1}\}$ and

$$s_k = \frac{r_k - P_{k-1}(z_k)}{Q_k(z_k)}. \tag{26}$$

Proof. Let $R_k = (r_i : i = 1, \cdots, k)$ be a given data vector, and let us consider P_{k-1} the interpolating polynomial for $R_{k-1} = (r_i : i = 1, \cdots, k-1)$ on Z_{k-1} in $\mathscr{P}_{k-1}$. For $i = 1, \cdots, k-1$, as $(I_1^{(k)}, \cdots, I_p^{(k)})$ is a (Z_{k-1}, z_k)-partition, then $Q_k(z_i) = 0$. So the polynomial P_k defined by the expression (25) above verifies $P_k(z_i) = P_{k-1}(z_i) = r_i$, $i = 1, \cdots, k-1$. On the other hand, we have by definition $x_i(z_k) \notin I_i^k$, $\forall i \in \{1, \cdots, p\}$; then, $Q_k(z_k) \neq 0$. So taking

$$s_k = \frac{r_k - P_{k-1}(z_k)}{Q_k(z_k)}, \tag{27}$$

we obtain $P_k(z_k) = r_k$. We conclude that

$$P_k(z_i) = r_i, \quad \forall i = 1, \cdots, k, \tag{28}$$

so P_k is an interpolating polynomial for R_k on Z_k in $\mathscr{P}_k = \text{span}\{Q_1, \cdots, Q_k\}$. We deduce that the linear mapping

$$P \in \mathscr{P}_k \mapsto (P(z_1), \cdots, P(z_k)) \in \mathbb{K}^{Z_k} \tag{29}$$

is surjective. But as $\dim \mathscr{P}_k \leq k$, we conclude that the mapping is a linear isomorphism and that $\mathscr{P}_k$ is an interpolation space for Z_k, hence the result. □

Remark 10.

(1) The interpolating polynomials P_k obtained by the previous theorem depend on the indexation choice of the nodes of Z_n

(2) For constructing a solution of $\mathscr{P}(Z_n)$, the following algorithm RLMVPIA chooses a random indexation of the interpolation nodes

```
Input: Interpolation set Z and interpolation values R
n = lenght(Z)
P_0=0
Z_0 = []
for k = 1 to n :
        z_k = a random point of Z
        Q_k = ZPNA(Z_{k-1}, z_k)[1]
        Z_k = Z_{k-1} ∪ {z_k}
        remove z_k from Z
        s_k = (r_k, −, P_{k-1}, (z_k))/Q_k(z_k)
        P_k = P_{k-1} + s_k Q_k
endfor
return P_n
```

Algorithm 2: RLMVPIA.

4. Examples

4.1. Examples for the Case $p = 2$. We will give two examples: the first one concerns the particular case where the interpolation set is a full grid. We will see that the RLMVPIA and the RMVPIA [5] are equivalent. The second one is for a random configuration.

4.1.1. Example 1: Grid Case. When the interpolation set is a full grid, RLMVPIA gives a similar result to the one obtained in [5, 7]. In the following example already studied in [5] where the set of nodes, $Z_n = Z_{(n_1+1)(n_2+1)}$ is presented in Figure 1.

We take

$$\begin{aligned} Z_{12} &= \{0, 1, 2\} \times \left\{0, 1, -\frac{1}{2}, \frac{1}{2}\right\}, \\ R_{12} &= \left\{1, 0, -2, -1, 1, \frac{1}{2}, -1, 1, 0, \frac{3}{2}, \frac{7}{3}, -4,\right\}, \end{aligned} \tag{30}$$

by applying the random RLMVPIA, several times; we obtain the same interpolating polynomial given in [5].

$$\begin{aligned} P = 1 - \frac{1}{2}x - \frac{1}{2}x^2 + 3y + \frac{71}{12}xy - \frac{21}{4}x^2y + \\ -3y^2 + \frac{107}{6}xy^2 - \frac{49}{6}x^2y^2 - 2y^3 - 20xy^3 + \frac{38}{3}x^2y^3. \end{aligned} \tag{31}$$

For a random configuration using the RLMVPIA, we obtain different solutions, as can be seen in the following example.

4.1.2. Example 2. In this example, we take the interpolation set (Figure 2)

$$Z_{11} = \{(1,-1), (1,0), (1,1), (1,2), (1,3), (1,4), (5,4), (6,4), (7,4), (8,4), (9,4)\}, \tag{32}$$

and for interpolation values

$$R_{11} = \{34, 34, -19, -16, -4, -18, -1, 24, 18, -27, -4\}, \tag{33}$$

by applying the random RLMVPIA several times; we give among the solutions obtained the two following ones. The first one

$$\begin{aligned} \text{Sol}_1 = \frac{79}{140}x^5 - \frac{359}{24}x^4 + \frac{3103}{21}x^3 - \frac{15961}{24}x^2 + \frac{548803}{420}x \\ + \frac{7}{5}y^5 - \frac{27}{2}y^4 + \frac{229}{6}y^3 - 13y^2 - \frac{991}{15}y - 741, \end{aligned} \tag{34}$$

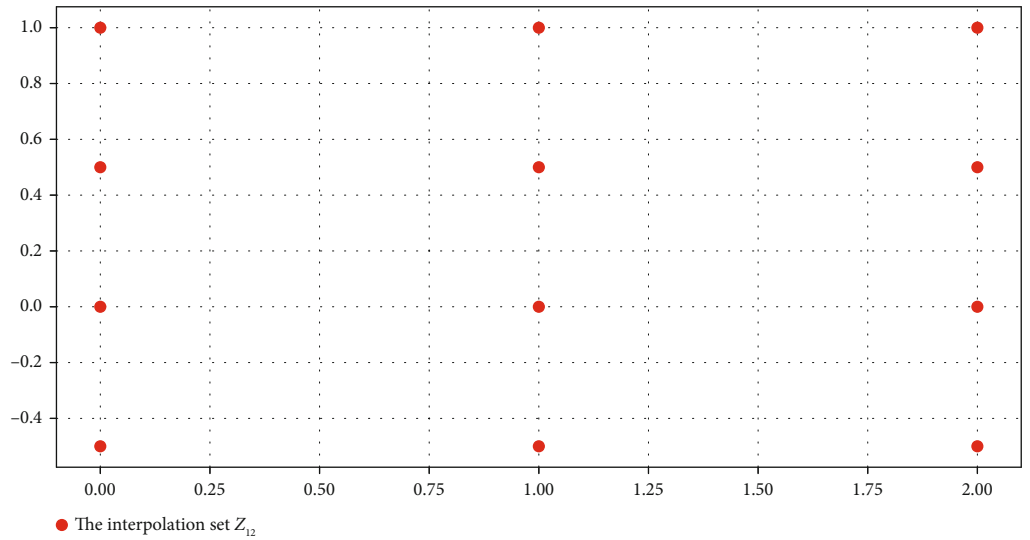

Figure 1: The interpolation set is a grid.

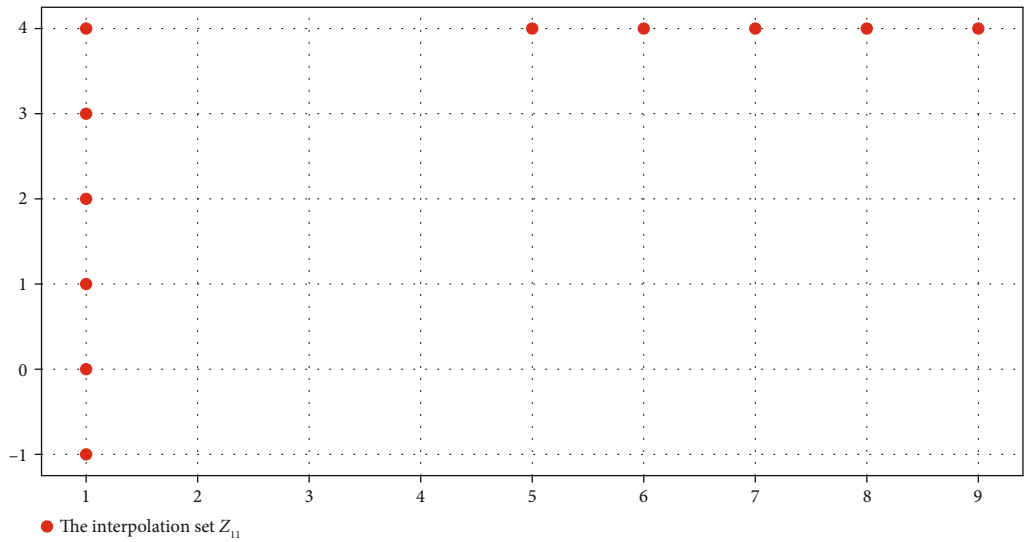

Figure 2: The interpolation set is a random configuration.

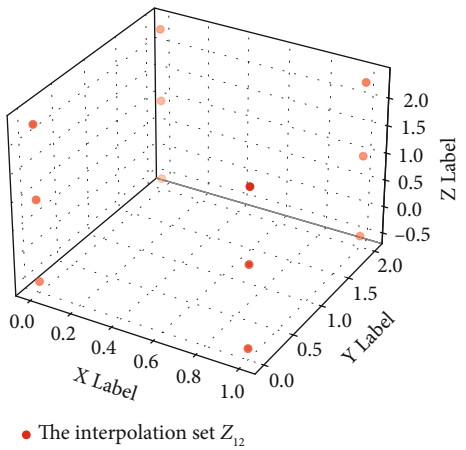

Figure 3: The interpolation set is a grid ($p = 3$).

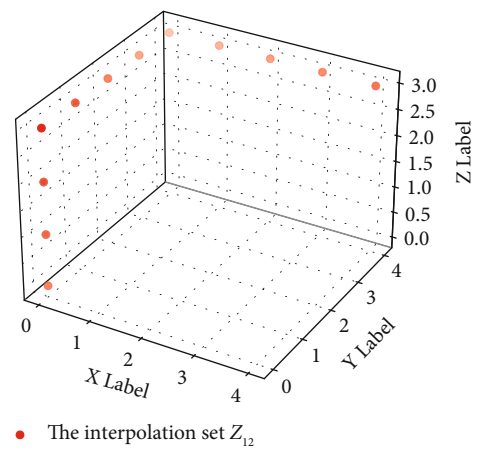

Figure 4: The interpolation set is a random configuration ($p = 3$).

and the second one

$$\begin{aligned}\mathrm{Sol}_2 = &-\frac{1}{4800}x^5y^5 + \frac{53}{40320}x^5y^4 - \frac{1}{630}x^5y^3 - \frac{53}{40320}x^5y^2 \\ &+ \frac{181}{100800}x^5y + \frac{187}{336}x^5 + \frac{7}{960}x^4y^5 - \frac{53}{1152}x^4y^4 \\ &+ \frac{203}{2880}x^4y^3 + \frac{53}{1152}x^4y^2 - \frac{7}{90}x^4y - \frac{187}{12}x^4 \\ &- \frac{97}{960}x^3y^5 + \frac{5141}{8064}x^3y^4 - \frac{21841}{20160}x^3y^3 - \frac{5141}{8064}x^3y^2 \\ &+ \frac{11939}{10080}x^3y + \frac{4559}{28}x^3 + \frac{133}{192}x^2y^5 - \frac{5035}{1152}x^2y^4 \\ &+ \frac{21607}{2880}x^2y^3 + \frac{5035}{1152}x^2y^2 - \frac{11801}{1440}x^2y - \frac{18547}{24}x^2 \\ &- \frac{1879}{800}xy^5 + \frac{103507}{6720}xy^4 - \frac{175969}{6720}xy^3 - \frac{103507}{6720}xy^2 \\ &+ \frac{918863}{33600}xy + \frac{531425}{336}x + \frac{63}{20}y^5 - \frac{201}{8}y^4 \\ &+ \frac{5555}{96}y^3 - \frac{11}{8}y^2 - \frac{41437}{480}y - \frac{7381}{8}.\end{aligned} \tag{35}$$

4.2. Examples for the Case p = 3. In this section, we also give two examples for a full grid and for a random configuration.

4.2.1. Example 3: Grid Case. In this example, we take the full grid of $\mathbb{R}^3$ considered in [5] (Figure 3)

$$Z_{12} = \{0, 1\} \times \{2, 0\} \times \left\{1, -\frac{1}{2}, \frac{7}{3}\right\}, \tag{36}$$

and the interpolation values

$$R_{12} = \left\{1, 0, -2, -1, 1, \frac{1}{2}, -1, 1, \frac{22}{7}, 0, \frac{9}{2}, -3\right\}. \tag{37}$$

by applying the random RLMVPIA, several times; we obtain the same interpolating polynomial given in [5].

$$\begin{aligned}P = &\frac{-943}{408} + \frac{1091}{408}x + \frac{8647}{5712}y - \frac{8899}{5712}y + \frac{139}{408}xz + \frac{3887}{5712}yz \\ &- \frac{671}{408}z - \frac{1283}{5712}xyz + \frac{133}{68}z^2 - \frac{137}{68}xz^2 - \frac{661}{952}yz^2 + \frac{745}{952}xyz^2,\end{aligned} \tag{38}$$

which coincides with the solution obtained by RMVPIA in [5].

4.2.2. Example 4. In this example, we consider another configuration of the interpolation set in $\mathbb{R}^3$ (Figure 4)

$$\begin{aligned}Z_{12} = &\{(0, 0, 0), (0, 0, 1), (0, 0, 2), (0, 0, 3), (0, 1, 3), (0, 2, 3), \\ &\cdot (0, 3, 3), (0, 4, 3), (1, 4, 3), (2, 4, 3), (3, 4, 3), (4, 4, 3)\},\end{aligned} \tag{39}$$

and we take them as interpolation values

$$R_{12} = \left\{0, 4, \frac{1}{3}, 1, -1, 7, \frac{4}{3}, 2, 3, \frac{1}{2}, -2, 5\right\}. \tag{40}$$

Applying the random RLMVPIA several times, we give among the solutions obtained the two following ones. The first one

$$\begin{aligned}\mathrm{Sol}_1 = &\frac{1}{576}x^4y^4z^3 - \frac{1}{192}x^4y^4z^2 + \frac{1}{288}x^4y^4z - \frac{1}{96}x^4y^3z^3 \\ &+ \frac{1}{32}x^4y^3z^2 - \frac{1}{48}x^4y^3z + \frac{11}{576}x^4y^2z^3 - \frac{11}{192}x^4y^2z^2 \\ &+ \frac{11}{288}x^4y^2z - \frac{1}{96}x^4yz^3 + \frac{1}{32}x^4yz^2 - \frac{1}{48}x^4yz \\ &- \frac{11}{1728}x^3y^4z^3 + \frac{11}{576}x^3y^4z^2 - \frac{11}{864}x^3y^4z \\ &+ \frac{11}{288}x^3y^3z^3 - \frac{11}{96}x^3y^3z^2 + \frac{11}{144}x^3y^3z - \frac{121}{1728}x^3y^2z^3 \\ &+ \frac{121}{576}x^3y^2z^2 - \frac{121}{864}x^3y^2z + \frac{11}{288}x^3yz^3 - \frac{11}{96}x^3yz^2 \\ &+ \frac{11}{144}x^3yz - \frac{1}{192}x^2y^4z^3 + \frac{1}{64}x^2y^4z^2 - \frac{1}{96}x^2y^4z \\ &+ \frac{1}{32}x^2y^3z^3 - \frac{3}{32}x^2y^3z^2 + \frac{1}{16}x^2y^3z - \frac{11}{192}x^2y^2z^3 \\ &+ \frac{11}{64}x^2y^2z^2 - \frac{11}{96}x^2y^2z + \frac{1}{32}x^2yz^3 - \frac{3}{32}x^2yz^2 \\ &+ \frac{1}{16}x^2yz + \frac{29}{1728}xy^4z^3 - \frac{29}{576}xy^4z^2 + \frac{29}{864}xy^4z \\ &- \frac{29}{288}xy^3z^3 + \frac{29}{96}xy^3z^2 - \frac{29}{144}xy^3z + \frac{319}{1728}xy^2z^3 \\ &- \frac{319}{576}xy^2z^2 + \frac{319}{864}xy^2z - \frac{29}{288}xyz^3 + \frac{29}{96}xyz^2 \\ &- \frac{29}{144}xyz + \frac{131}{432}y^4z^3 - \frac{131}{144}y^4z^2 + \frac{131}{216}y^4z - \frac{535}{216}y^3z^3 \\ &+ \frac{535}{72}y^3z^2 - \frac{535}{108}y^3z + \frac{2653}{432}y^2z^3 - \frac{2653}{144}y^2z^2 \\ &+ \frac{2653}{216}y^2z - \frac{929}{216}yz^3 + \frac{929}{72}yz^2 - \frac{929}{108}yz \\ &+ 2z^3 - \frac{59}{6}z^2 + \frac{71}{6}z.\end{aligned} \tag{41}$$

The second one with a lower degree

$$\begin{aligned}\mathrm{Sol}_2 = &\frac{1}{4}x^4 - \frac{11}{12}x^3 - \frac{3}{4}x^2 + \frac{29}{12}x + \frac{131}{72}y^4 + \frac{1}{8}y^3z - \frac{1097}{72}y^3 \\ &- \frac{7}{8}y^2z + \frac{1421}{36}y^2\frac{1}{2}yz^3 + 3yz^2 - \frac{15}{4}yz - \frac{505}{18}y \\ &+ 2z^3 - \frac{59}{6}z^2 + \frac{71}{6}z.\end{aligned} \tag{42}$$

Remark 11. The last example shows the importance of the random approach, in some cases, to get solutions of fairly low degree, which can influence the cost of evaluations.

5. Conclusion

In conclusion, this work contributes to solve the problem of Lagrange multivariate polynomial interpolation with any finite set of interpolation nodes, using a recursive algorithm RLMVPIA with a random approach based on the (Z, z) -partition concept. This study shows that RLMVPIA is easy to implement and requires no storage.

Currently, we are interested firstly, in refining the random approach of the algorithm, to build, in a more deterministic way, optimal solutions of smaller degree. The problem of optimal solution and the study of the numerical stability of the RLMVPIA are under investigation. One can also find natural applications of RLMVPIA in different topics of applied mathematics and engineering as the numerical resolution of PDEs, computer-aided design (CAD), cryptography, etc.

Data Availability

Our research does not use any archived datasets.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

We are grateful to our Professor A. Messaoudi for his help and support.

References

[1] J. Carnicer and T. Sauer, "Leibniz rules for multivariate divided differences," *Journal of Approximation Theory*, vol. 181, pp. 43–53, 2014.

[2] M. Gasca and T. Sauer, "On the history of multivariate polynomial interpolation," *Journal of Computational and Applied Mathematics*, vol. 122, no. 1-2, pp. 23–35, 2000.

[3] T. Sauer, "Polynomial interpolation of minimal degree," *Numerische Mathematik*, vol. 78, no. 1, pp. 59–85, 1997.

[4] I. S. Berezin and N. P. Zhidkov, *Computing methods*, Addison-Wesley, Reading, MA, 1959.

[5] M. Errachid, A. Essanhaji, and A. Messaoudi, "RMVPIA: a new algorithm for computing the Lagrange multivariate polynomial interpolation," *Numerical Algorithms Journal*, vol. 84, no. 4, pp. 1507–1534, 2020.

[6] E. Isaacson and H. B. Keller, *Analysis of Numerical Methods*, Wiley, New York, 1966.

[7] R. D. Neidinger, "Multivariate polynomial interpolation in Newton forms," *SIAM Review*, vol. 61, no. 2, pp. 361–381, 2019.

[8] T. Sauer, "Lagrange interpolation on subgrids of tensor product grids," *Mathematics of Computation*, vol. 73, no. 245, pp. 181–190, 2004.

[9] J. Czekansky and T. Sauer, "The multivariate Horner scheme revisited," *BIT*, vol. 55, no. 4, pp. 1043–1056, 2015.

[10] M. Errachid, K. Jbilou, A. Messaoudi, and H. Sadok, "GRPIA: a new algorithm for computing interpolation polynomials," *Numerical Algorithms Journal*, vol. 80, pp. 253–278, 2019.

[11] A.-J. Macleod, "A comparison of algorithms for polynomial interpolation," *Journal of Computational and Applied Mathematics*, vol. 8, no. 4, pp. 275–277, 1982.

[12] J. M. Peña and T. Sauer, "On the multivariate Horner scheme," *SIAM Journal on Numerical Analysis*, vol. 37, no. 4, pp. 1186–1197, 2000.

[13] T. Sauer, "Computational aspects of multivariate polynomial interpolation," *Advances in Computational Mathematics*, vol. 3, no. 3, pp. 219–237, 1995.

[14] T. Sauer and Y. Xu, "A case study in multivariate Lagrange interpolation," in *Approximation Theory Wavelets and Applications, Maratea, (1994), NATO Adv. Sci. Inst. Ser. C Math. Phys. Sci. 454*, pp. 443–452, Kluwer Academic, Dordrecht, the Netherlands, 1995.

[15] H. Werner, "Remarks on Newton type multivariate interpolation for subsets of grids," *Computing*, vol. 25, no. 2, pp. 181–191, 1980.

[16] C. de Boor and A. Ron, "Computational aspects of polynomial interpolation in several variables," *Mathematics of Computation*, vol. 58, no. 198, pp. 705–727, 1992.

[17] N. Dyn and M. S. Floater, "Multivariate polynomial interpolation on lower sets," *Journal of Approximation Theory*, vol. 177, pp. 34–42, 2014.

[18] M. Gasca and T. Sauer, "Polynomial interpolation in several variables," *Advances in Computational Mathematics*, vol. 12, pp. 377–410, 2000.

[19] F. Dell'accio, F. Di Tommaso, and N. Siar, "On the numerical computation of bivariate Lagrange polynomials," *Applied Mathematics Letters*, vol. 112, p. 106845, 2021.

[20] F. Dell'Accio, F. Di Tommaso, O. Nouisser, and N. Siar, "Solving Poisson equation with Dirichlet conditions through multinode Shepard operators," *Computers and Mathematics with Applications*, vol. 98, pp. 254–260, 2021.

[21] F. Dell'Accio, F. di Tommaso, N. Siar, and M. Vianello, "Numerical differentiation on scattered data through multivariate polynomial interpolation," *BIT Numerical Mathematics*, 2021.

[22] G. C. Meletiou, D. K. Tasoulis, and M. N. Vrahtis, "Cryptography through interpolation, approximation and computational intelligence methods," *Bulletin of the Greek Mathematical Society*, vol. 48, pp. 61–75, 2003.