

Basis-Independent Polynomial Division Algorithm Applied to Division in Lagrange and Bernstein Basis

Manfred Minimair**

Department of Mathematics and Computer Science
Seton Hall University, 400 South Orange Avenue
South Orange, New Jersey 07079, USA
manfred@minimair.org, <http://minimair.org>

Abstract. Division algorithms for univariate polynomials represented with respect to Lagrange and Bernstein basis are developed. These algorithms are obtained by abstracting from the classical polynomial division algorithm for polynomials represented with respect to the usual power basis. It is shown that these algorithms are quadratic in the degrees of their inputs, as in the power basis case.

Keywords: polynomial division, Lagrange basis, Bernstein basis

1 Introduction

Fundamental operations for polynomials represented with respect to bases other than the usual power basis are being intensely studied. Examples include computation of resultants and resultant matrices [16, 22, 8], gcds [17, 12, 10], generalized companion matrices [7, 23, 24, 11], polynomial remainder sequences [6, 14], and polynomial division [5, 19, 1]. Carrying out fundamental operations over alternative bases is motivated by the desire to avoid computational cost and numeric errors incurred by converting between different polynomial bases [9, 15, 20] and practical applications [3, 4].

The current paper studies division of univariate polynomials represented with respect to the Lagrange basis as well as the Bernstein basis. Current algorithms for polynomial division over these bases ([1, 2] and respectively [19]) proceed by setting up systems of linear equations for the coefficients of the quotient and remainder. Then solutions are computed by respectively using SVD and LU factorization. Since the sizes of these systems of equations are linear in the degrees of the input polynomials and the solution methods require a worst-case cubic number of arithmetic operations the worst-case complexity of these methods is expected to be cubic. However, it is well-known that the worst-case complexity of dividing polynomials represented with respect to the usual power basis is quadratic. For example, [21] shows that for polynomials represented in power basis division by the classical division algorithm is $O((n+1)(m-n+1))$, where

** Supported by the NSF grant CCF 0430741

m and n respectively is the degree of the dividend and divisor. Therefore there seems to be a gap in the current theory. It seems that an appropriate generalization of the classical division algorithm that exhibits quadratic worst-case complexity for Lagrange and Bernstein basis is lacking. The current paper addresses this gap by providing a generalized division algorithm (Section 2) and by showing quadratic worst-case complexities for the number of arithmetic operations required for division over Lagrange (Theorem 15) and Bernstein (Theorem 26) basis.

Before we conclude this introductory section, we discuss some other related works, besides [1, 2, 19]. The work [5] uses matrix techniques to divide polynomials over orthogonal bases which obviously do not include the Lagrange and Bernstein bases. Therefore these techniques seem not immediately applicable to dividing polynomials over Lagrange and Bernstein basis. These techniques are based on the Cayley-Hamilton Theorem and require generalized companion matrices, the comrade matrices. But recently, some generalized companion matrices over the Lagrange and Bernstein basis have been developed [11, 23, 24]. It would be interesting to attempt to generalize the techniques from [5] to these companion matrices. However it is not obvious if a resulting matrix-based division algorithm would be of quadratic worst-case complexity. In the case of the Bernstein basis it may even be doubtful because of the cost associated with computing the generalized companion matrix [24]. Another approach for addressing the problem of the current paper would be to efficiently convert the polynomials into power basis [9, 15, 20], to apply some fast polynomial division algorithm in power basis and to convert the result back into Lagrange or Bernstein basis. However, this is not the focus of the current paper. The goal of the current paper is to provide an algorithm that does not require basis conversion.

The paper is organized as follows. Section 2 provides a generalized (basis-independent) framework for polynomial division. The following two sections, 3 and 4, apply the generalized framework to polynomials represented respectively in Lagrange and Bernstein basis.

2 Basis-independent framework for polynomial division

The purpose of this section is to give a general framework, which is basis-independent, for the well-known classical division algorithm for polynomials given in power basis representation. In subsequent sections this framework will be specialized in order to obtain division algorithms for polynomials in Lagrange and in Bernstein basis.

2.1 Motivating example

We divide the polynomial f_0 by the polynomial g presented in power basis by $f_0 = 6x^3 + 3x^2 + 12x - 3$ and $g = 2x + 1$. When carrying out the division we generate a sequence f_1, f_2, f_3 where f_{i+1} is obtained from f_i by removing the

leading term, that is,

$$\begin{aligned} f_1 &= f_0 - 3x^2g = 0x^2 + 12x - 3 \\ f_2 &= f_1 = 12x - 3 \\ f_3 &= f_2 - 6g = -9. \end{aligned}$$

In this sequence each f_{i+1} has a smaller degree than its successor f_i . Notice that f_1 and f_3 are formed by subtracting multiples of g from f_0 and respectively f_2 such that the respective leading terms $6x^3$ and $12x$ vanish. In the case of f_2 no subtraction has to be carried out because $0x^2$, the leading term of f_1 , contains a vanishing coefficient. Moreover, the sequence of f_i 's stops with f_3 because the degree of f_3 is less than the degree of g . Obviously f_3 is the remainder of f_0 divided by g .

According to the sequence of f_i 's we can also form a sequence of partial quotients $q_1 = 3x^2$, $q_2 = 0x$, $q_3 = 6$. Then the quotient of f_0 divided by g is the sum $q_1 + q_2 + q_3$ of the partial quotients.

So, what are the key operations needed for carrying out the division?

1. Extracting the coefficient of the leading term of a polynomial, e.g. when forming f_1 and f_3 . Subsequently, we will denote this operation with the symbol Head.
2. Extracting the part of a polynomial minus the leading term, e.g. when forming f_2 . Subsequently, we will denote this operation with the symbol Tail.
3. Multiple of g matching the degree of f_i , e.g. x^2g . Subsequently, we will denote this operation with the symbol Match.
4. Partial quotient, the multiplier for the operation Match, e.g. x^2 for x^2g . Subsequently, we will denote this operation with the symbol Quot.

2.2 Definition of polynomial division algorithm

We define a polynomial division algorithm in a general setting independent from the bases in which the polynomials are represented. The algorithms uses some basic operators we define first.

Definition 1 For some field F , the symbol $F[x]_m$ denotes the F -vector space of polynomials of degree up to m in the variable x . Moreover we let $\deg(0) = -1$ and for $m < 0$ the symbol $F[x]_m$ denotes the (trivial) F -vector space $\{0\}$.

Definition 2 (Basic operators)

Head: By $\text{Head}_m(f)$ we denote the coefficient of x^m of the polynomial f in $F[x]_m$.

Tail: By Tail_m we denote a function that identically maps any polynomial f in $F[x]_m$ of degree at most $m - 1$ into $F[x]_{m-1}$.

Match: If f, g respectively is a polynomial of degree m and n , with $n \leq m$, then $\text{Match}_k(f, g) = qg$ for some polynomial $q \in F[x]_k$ of degree $m - n \leq k$.

Quot: $\text{Quot}_k(f, g)$ denotes the factor q in the definition of $\text{Match}_k(f, g)$.

The operator Tail in the above definition is important in computations. That is, if f is represented with respect to some basis of $F[x]_m$, then $\text{Tail}_m(f)$ is f represented with respect to some basis of $F[x]_{m-1}$.

In the above definition of the operator Match, the polynomial q is only specified up to degree. Later, it will be specified precisely as suitable for computations in Lagrange and Bernstein basis. The objective of its definition will be to allow to efficiently compute the product $\text{Match}_{m-n}(f, g) = \text{Quot}_{m-n}(f, g) \cdot g$. Moreover note that the operator Match only depends on the degree of f , not on its coefficients. We included f in the definition rather than only its degree for the sake of a clearer presentation.

The subsequent examples illustrate the definitions for polynomials represented in the power basis $\{1, x, x^2, x^3, \dots\}$.

Example 3 For polynomials in power basis representation,

1. $\text{Head}_m(\sum_{k=0}^m a_k x^k) = a_m$,
2. $\text{Tail}_m(0 \cdot x^m + \sum_{k=0}^{m-1} a_k x^k) = \sum_{k=0}^{m-1} a_k x^k$,
3. $\text{Match}_{m-n}(f, g) = x^{m-n} g$, where f, g respectively is of degree m and n ,
4. $\text{Quot}_{m-n}(f, g) = x^{m-n}$.

Polynomial division algorithm

Input: polynomials $f \in F[x]_m$ and $g \in F[x]_n$, where g had degree $n \leq m$.

Output: the quotient Q and the remainder R of f divided by g

Let $f_0 = f$. Generate sequences of polynomials f_i and q_i , for $i = 1, \dots, m-n+1$, such that

$$f_{i+1} = \text{Tail}_{m-i}(f_i - \frac{\text{Head}_{m-i}(f_i)}{\text{Head}_{m-i}(\text{Match}_{m-n}(f_i, g))} \text{Match}_{m-n}(f_i, g)),$$

$$q_{i+1} = \frac{\text{Head}_{m-i}(f_i)}{\text{Head}_{m-i}(\text{Match}_{m-n}(f_i, g))} \text{Quot}_{m-n}(f_i, g).$$

Then $Q = q_1 + \dots + q_{m-n+1}$ and $R = f_{m-n+1}$.

Remark 4 It can happen that $\text{Head}(f_i) = 0$ (see the motivating example above). In this case the definition of f_{i+1} simply is $\text{Tail}_{m-i}(f_i)$ and respectively q_{i+1} is zero.

2.3 Complexity of the division algorithm

We investigate the worst-case complexity of the number of arithmetic operations required by the polynomial division algorithm.

For the remainder of this section we assume that the polynomials are represented appropriately in order to guarantee the expected linear complexity of addition, subtraction and multiplication by a constant. This is obviously possible if a polynomial in $F[x]_m$ is represented by the list of $(m+1)$ coefficients with respect to a fixed basis. Therefore we make the following assumption.

Assumption 5 *The polynomials $f, g \in F[x]_m$ are represented such that the number of arithmetic operations needed for computing $f \pm g$ and $c \cdot f$, for any constant c , is $O(m)$.*

Next we formulate some natural assumptions on the complexities for computing the basic operators from Definition 2, which are satisfied for computations in the standard power basis. Later we will investigate whether the basic operators for the Lagrange and Bernstein bases fulfill these assumptions.

Assumption 6 *The number of arithmetic operations needed for computing*

$$\begin{array}{ll} \text{Head}_m(f) \text{ is } O(m), & \text{Match}_k(f, g) \text{ is } O(n), \\ \text{Tail}_m(f) \text{ is } O(m), & \text{Quot}_k(f, g) \text{ is } O(k). \end{array}$$

Theorem 7 *Under Assumption 6 the number of arithmetic operations needed for computing the quotient and remainder of f by g is $O(m \cdot (m - n + 1))$.*

Proof: Since the division algorithm generates $m - n + 1$ polynomials f_i and q_i . It remains to check that the number of arithmetic operations is $O(m)$ for each i . Notice that by the definition of the operator Tail the degree of f_i is at most $m - i$. Therefore $\frac{\text{Head}(f_i)}{\text{Head}(\text{Match}_{m-n}(f_i, g))}$ is $O(m)$. Thus computing f_{i+1} and q_{i+1} is $O(m)$. Furthermore, computing the sum in the quotient Q is $O((m - n)(m - n + 1))$ which is $O(m \cdot (m - n + 1))$. \square

Remark 8 By [21] division over the power basis is $O(n \cdot (m - n + 1))$. Notice that the left-hand factor only depends on n and not on m . This is due to being able to retrieve the leading coefficient of a polynomial represented in power basis in constant time. For Lagrange and Bernstein basis this operation is non-constant, $O(m)$. Hence we get the factor m in the complexity.

Next we prove the correctness of the division algorithm.

2.4 Correctness of the division algorithm

The sequence of f_i 's satisfies the invariant $f_i = q_{i+1}g + f_{i+1}$. Thus $f_0 = (\sum_{i=1}^{m-n+1} q_i)g + f_{m-n+1}$. Since by the definition of the operator Tail the degrees of the polynomials of the sequence f_i are decreasing, the degree of f_{m-n+1} is less than the degree of g . Therefore $Q = \sum_{i=1}^{m-n+1} q_i$ is the quotient of the division of f_0 by g and $R = f_{m-n+1}$ is the remainder.

3 Division in Lagrange basis

This section consists of three parts. The first part defines the basic operators required for polynomial division (Definition 2). The second part derives the complexity of the division algorithm. The third part proves the correctness of the definitions.

3.1 Definition of basic operators

We start with the definition of the Lagrange basis.

Definition 9 (Lagrange basis) Let $\lambda_{j,m} = \frac{\pi_{j,m}}{\bar{\pi}_{j,m}}$ where $\pi_{j,m} = \prod_{\substack{i=0 \\ i \neq j}}^m (x - x_i)$ and $\bar{\pi}_{j,m} = \prod_{\substack{i=0 \\ i \neq j}}^m (x_j - x_i)$. Then $\lambda_{m,m}, \dots, \lambda_{0,m}$ are called the Lagrange basis of degree m .

Definition 10 We define the head and tail operators as

$$\begin{aligned} \text{Head}_m\left(\sum_{j=0}^m a_j \lambda_{j,m}\right) &= \sum_{j=0}^m \frac{a_j}{\bar{\pi}_{j,m}}, \\ \text{Tail}_m\left(\sum_{j=0}^m a_j \lambda_{j,m}\right) &= \sum_{j=0}^{m-1} a_j \lambda_{j,m-1}. \end{aligned}$$

Definition 11 We define the match and quot operators as

$$\begin{aligned} \text{Match}_k\left(f, \sum_{j=0}^n b_j \lambda_{j,n}\right) &= \sum_{j=0}^n b_j \bar{\pi}_{j,m} \bar{\pi}_{j,n}^{-1} \lambda_{j,m}, \\ \text{Quot}_k\left(f, \sum_{j=0}^n b_j \lambda_{j,n}\right) &= \sum_{j=0}^u \bar{\pi}_{j,m} \bar{\pi}_{j,n}^{-1} \lambda_{j,k}, \end{aligned}$$

where $f \in F[x]_m$ and $u = \min(k, n)$, if $m > n$, and $u = k$, if $m = n$.

The essential properties of the match and quot operators are

$$\begin{aligned} \text{Match}_k(f, g) &= \text{Quot}_k(f, g) \cdot g, \\ \text{Quot}_k(f, g) &= \prod_{i=n+1}^m (x - x_i) \end{aligned}$$

which will be shown in the section on the correctness of the operators below. Notice that $\text{Quot}_k(f, g)$ has been chosen such that the product $\text{Quot}_k(f, g) \cdot g = \text{Match}_k(f, g)$ can be easily computed in the basis for $F[x]_m$.

Next we give an example for polynomial division when using these operators. We use the same polynomials as in Section 2.1.

Example 12 With $x_0 = 0, x_1 = 1, x_2 = 2, x_3 = 3$, let the polynomials $f_0 = 222 \lambda_{3,3} + 81 \lambda_{2,3} + 18 \lambda_{1,3} - 3 \lambda_{0,3}$ and $g = 3 \lambda_{1,1} + 1 \lambda_{0,1}$. Then

$$\begin{aligned} f_1 &= f_0 - 3(-2x(x-2) + 3(x-1)(x-2))g = f_0 - 3(2\lambda_{1,2} + 6\lambda_{0,2})g \\ &= 81\lambda_{2,2} - 21\lambda_{0,2}, \\ f_2 &= f_1 - 15(-(x-1)(x-2) + x(x-2))g = f_1 - 15(-1\lambda_{1,2} - 2\lambda_{0,2})g \\ &= 45\lambda_{1,1} + 9\lambda_{0,1} \\ f_3 &= f_2 - 18 \cdot 1 \cdot g = f_2 - 18(\lambda_{2,2} + \lambda_{1,2} + \lambda_{0,2})g \\ &= -9\lambda_{0,0} \end{aligned}$$

3.2 Complexity

We investigate the number of arithmetic operations required to carry out the polynomial division algorithm. Before we state the main result, Theorem 15, we give some auxiliary lemmas.

Lemma 13 *The number of arithmetic operations required to compute*

$$(\overline{\pi}_{0,n}, \overline{\pi}_{0,n+1}, \dots, \overline{\pi}_{0,m}), \dots, (\overline{\pi}_{m,n}, \overline{\pi}_{m,n+1}, \dots, \overline{\pi}_{m,m}) \text{ is } O(m^2). \quad (1)$$

Proof: Since $\overline{\pi}_{j,i+1} = (x_j - x_{i+1}) \cdot \overline{\pi}_{j,i}$, for $j \leq i$ or $j > i+1$ and $\overline{\pi}_{j,i+1} = \overline{\pi}_{j,i}$, for $j = i+1$, computing $\overline{\pi}_{j,n}, \overline{\pi}_{j,n+1}, \dots, \overline{\pi}_{j,m}$ is $O(m)$ for each $j = 0, \dots, m$. \square

Lemma 14 *Given all required values of $\overline{\pi}_{j,i}$, the basic operators from Definitions 11 and 10 satisfy Assumption 6.*

Proof: The head operator requires computing the sum of $\frac{a_j}{\overline{\pi}_{j,m}}$ for $j = 0, \dots, m$ which is $O(m)$. Furthermore, the tail operator is $O(1)$.

The match and quot operators require computing $\overline{\pi}_{j,m}^{-1} \overline{\pi}_{j,n}$ either for indices $j = 0, \dots, \min(k, n)$ for $\text{Quot}_k(f, g)$ and for $j = 0, \dots, n$ for $\text{Match}_k(f, g)$, where $g \in F[x]_n$, which is $O(k)$ and respectively $O(n)$. Moreover $\text{Match}_k(f, g)$ requires computing $(\overline{\pi}_{j,m}^{-1} \overline{\pi}_{j,n}) \cdot b_j$ for $j = 0, \dots, n$ which is $O(n)$. Thus $\text{Match}_k(f, g)$ and $\text{Quot}_k(f, g)$ respectively is $O(n)$ and $O(k)$. \square

Now we are ready to show the complexity of division.

Theorem 15 *The number of arithmetic operations required for division in the Lagrange basis is $O(m^2)$.*

Proof: We observe that $\text{Match}_k(f, \sum_{j=0}^n a_j \lambda_{j,n})$, $\text{Quot}_k(f, \sum_{j=0}^n b_j \lambda_{j,n})$ and $\text{Head}_m(\sum_{j=0}^m a_j \lambda_{j,m})$ require the constants $\overline{\pi}_{j,n}$ and $\overline{\pi}_{j,m}$, for $j = 0, \dots, m$. Thus, considering $n \leq m$, by Lemma 13, and Theorem 7, the division algorithm is $O(m^2 + m(m - n + 1))$ which is $O(m^2)$. \square

3.3 Correctness

We prove the correctness of the definitions of the basic operators in the previous section. The correctness follows from two theorems, 16 and 19, which respectively verify the defining properties of the head/tail and match/quot operators.

Theorem 16 *If $\sum_{j=0}^m \frac{a_j}{\overline{\pi}_{j,m}} = 0$ then $\sum_{j=0}^m a_j \lambda_{j,m} = \sum_{j=0}^{m-1} a_j \lambda_{j,m-1}$. Furthermore, $\sum_{j=0}^m \frac{a_j}{\overline{\pi}_{j,m}}$ is the coefficient of x^m in the polynomial $\sum_{j=0}^m a_j \lambda_{j,m}$.*

The right-hand side of the above equality is used to define the tail operator and that the coefficient of x^m is returned by the head operator.

Proof: Notice that the leading coefficient of $\lambda_{j,m}$ is $\frac{1}{\overline{\pi}_{j,m}}$. Therefore the coefficient of x^m in $f = \sum_{j=0}^m a_j \lambda_{j,m}$ is $\sum_{j=0}^m \frac{a_j}{\overline{\pi}_{j,m}}$. If $\sum_{j=0}^m \frac{a_j}{\overline{\pi}_{j,m}} = 0$ then

the degree of f is at most $m - 1$. In this case m interpolation points, say, $(x_0, a_0), \dots, (x_{m-1}, a_{m-1})$ are sufficient to determine f . Therefore, we get $f = \sum_{j=0}^{m-1} a_j \lambda_{j,m-1}$. \square

Before proving the next theorem we give some auxiliary lemmas.

The following observation allows us to relate $\lambda_{j,m}$ to $\lambda_{j,n}$.

Lemma 17 $\lambda_{j,m} = \prod_{i=n+1}^m (x - x_i) \overline{\pi}_{j,m}^{-1} \overline{\pi}_{j,n} \lambda_{j,n}$ if $m \geq n$ and $j \leq n$.

Proof: For $j \leq n$,

$$\lambda_{j,m} = \overline{\pi}_{j,m}^{-1} \prod_{\substack{i=1 \\ i \neq j}}^n (x - x_i) \prod_{i=n+1}^m (x - x_i) = \overline{\pi}_{j,m}^{-1} \overline{\pi}_{j,n} \prod_{i=n+1}^m (x - x_i) \lambda_{j,n}.$$

\square

The next lemma allows us to write the factor from Lemma 17 that relates $\lambda_{j,m}$ to $\lambda_{j,n}$ in terms of the Lagrange basis.

Lemma 18

$$\prod_{i=n+1}^m (x - x_i) = \sum_{j=0}^u \overline{\pi}_{j,m} \overline{\pi}_{j,n}^{-1} \lambda_{j,k},$$

where $u = \min k, n$, if $0 < m - n \leq k$, and $u = k$, if $m = n$.

Proof: Interpolating $\prod_{i=n+1}^m (x - x_i)$ for $x = x_0, \dots, x_k$, yields

$$\prod_{i=n+1}^m (x - x_i) = \sum_{j=0}^k \left(\prod_{i=n+1}^m (x_j - x_i) \right) \lambda_{j,k}.$$

In the trivial case of $m = n$ the products for i ranging from $n + 1$ to m are 1, and thus the lemma holds for this case. So, let us assume the other case $0 < m - n \leq k$. Observe that $\prod_{i=n+1}^m (x_j - x_i) = 0$ for $j \geq n + 1$. Therefore,

$$\prod_{i=n+1}^m (x - x_i) = \sum_{j=0}^{\min(k,n)} \left(\prod_{i=n+1}^m (x_j - x_i) \right) \lambda_{j,k}.$$

Furthermore, for $j \leq n$,

$$\prod_{i=n+1}^m (x_j - x_i) = \prod_{i=n+1}^m (x_j - x_i) \cdot \frac{\prod_{\substack{i=1 \\ i \neq j}}^n (x_j - x_i)}{\prod_{\substack{i=1 \\ i \neq j}}^n (x_j - x_i)} = \frac{\prod_{\substack{i=1 \\ i \neq j}}^m (x_j - x_i)}{\prod_{\substack{i=1 \\ i \neq j}}^n (x_j - x_i)} = \frac{\overline{\pi}_{j,m}}{\overline{\pi}_{j,n}}.$$

\square

The following theorem determines a multiple of a polynomial represented in Lagrange basis. This multiple is denoted by the operator Match and the factor by the operator Quot in the division algorithm.

Theorem 19

$$\sum_{j=0}^n b_j \bar{\pi}_{j,m} \bar{\pi}_{j,n}^{-1} \lambda_{j,m} = \left(\sum_{j=0}^u \bar{\pi}_{j,m} \bar{\pi}_{j,n}^{-1} \lambda_{j,k} \right) \left(\sum_{j=0}^n b_j \lambda_{j,n} \right),$$

where $u = \min(k, n)$, if $0 < m - n \leq k \leq m$, and $u = k$, if $m = n$.

Proof: The case $m = n$ is obvious because the left factor on the right-hand side of the equality is 1. So, let us consider the other case. By Lemmas 17 and 18,

$$\begin{aligned} \left(\sum_{j=0}^{\min(k,n)} \bar{\pi}_{j,m} \bar{\pi}_{j,n}^{-1} \lambda_{j,k} \right) \left(\sum_{j=0}^n b_j \lambda_{j,n} \right) &= \prod_{i=n+1}^m (x - x_i) \left(\sum_{j=0}^n b_j \lambda_{j,n} \right) \\ &= \left(\sum_{j=0}^n b_j \prod_{i=n+1}^m (x - x_i) \lambda_{j,n} \right) = \left(\sum_{j=0}^n b_j \bar{\pi}_{j,m} \bar{\pi}_{j,n}^{-1} \lambda_{j,m} \right). \end{aligned}$$

□

4 Division in Bernstein basis

This section consists of three parts. The first part defines the basic operators required for polynomial division (Definition 2). The second part derives the complexity of the division algorithm. The third part proves the correctness of the definitions of the basic operators.

4.1 Definition of basic operators

We start with the definition of the Bernstein basis.

Definition 20 (Bernstein basis) Let $\beta_{j,m} = \binom{m}{j} x^j (1-x)^{m-j}$. Then the polynomials $\beta_{m,m}, \dots, \beta_{0,m}$ are called the Bernstein basis of degree m .

Definition 21 We define the head and tail operators as

$$\begin{aligned} \text{Head}_m \left(\sum_{j=0}^m a_j \beta_{j,m} \right) &= \sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j, \\ \text{Tail}_m \left(\sum_{j=0}^m a_j \beta_{j,m} \right) &= \sum_{j=0}^{m-1} (-1)^j \left(\sum_{i=0}^j (-1)^i \binom{m}{i} a_i \right) \binom{m-1}{j}^{-1} \beta_{j,m-1}. \end{aligned}$$

Definition 22 We define the match and quot operators as

$$\begin{aligned} \text{Match}_k \left(f, \sum_{j=0}^n b_j \beta_{j,n} \right) &= \sum_{j=0}^n b_j \binom{n}{j} \binom{m}{j}^{-1} \beta_{j,m}, \\ \text{Quot}_k \left(f, \sum_{j=0}^n b_j \beta_{j,n} \right) &= \sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} \binom{k}{j}^{-1} \beta_{j,k}. \end{aligned}$$

The essential properties of the match and quot operators are

$$\begin{aligned}\text{Match}_k(f, g) &= \text{Quot}_k(f, g) \cdot g, \\ \text{Quot}_k(f, g) &= (1 - x)^{m-n}\end{aligned}$$

which will be shown in the section on the correctness of the operators below. Notice that $\text{Quot}_k(f, g)$ has been chosen such that the product $\text{Quot}_k(f, g) \cdot g = \text{Match}_k(f, g)$ can be easily computed in the basis for $F[x]_m$.

Next we give an example for polynomial division when using these operators. We use the same polynomials as in Section 2.1.

Example 23 Let

$$\begin{aligned}f &= 18 \beta_{3,3} + 6 \beta_{2,3} + 1 \beta_{1,3} - 3 \beta_{0,3} \\ g &= 3 \beta_{1,1} + 1 \beta_{0,1}\end{aligned}$$

Then

$$\begin{aligned}f_1 &= f_0 - 3(1-x)^2 g = f_0 - 3 \beta_{0,2} g = 18 \beta_{2,2} - 6 \beta_{0,2}, \\ f_2 &= f_1 - (-6)(1-x) g = f_1 - (-6) \left(\frac{1}{2} \beta_{1,2} + \beta_{0,2}\right) g = 18 \beta_{1,1}, \\ f_3 &= f_2 - 9 \cdot 1 \cdot g = f_2 - 9(\beta_{2,2} + \beta_{1,2} + \beta_{0,2}) g = -9 \beta_{0,0}.\end{aligned}$$

4.2 Complexity

We investigate the number of arithmetic operations required to carry out the polynomial division algorithm. Before we state the main result, Theorem 26, we give some auxiliary lemmas.

Lemma 24 *The number of arithmetic operations required to compute all*

$$\binom{i}{j}, \quad \text{for } j = 0, \dots, i \text{ and for } i = \min(m-n, n-1), \dots, m \quad (2)$$

is $O(m^2 - \min(m-n, n-1)^2)$.

Proof: Since $\binom{i}{j+1} = \frac{i!}{(i-j-1)!(j+1)!} = \frac{i-j}{j+1} \binom{i}{j}$, computing $\binom{i}{0}, \dots, \binom{i}{i}$ is $O(i)$. Thus computing $\binom{i}{j}$, for all $j = 0, \dots, i$ and for all $i = \min(m-n, n-1), \dots, m$ is of order $\sum_{i=\min(m-n, n-1)}^m i$ which is

$$O((m + \min(m-n, n-1))(m - \min(m-n, n-1) + 1)),$$

that is, $O(m^2 - \min(m-n, n-1)^2)$. \square

Lemma 25 *Given all required values of $\binom{i}{j}$, the basic operators from Definitions 22 and 21 satisfy Assumption 6.*

Proof: The head operator requires computing the sum of $(-1)^{m-j} \binom{m}{j} a_j$ for $j = 0, \dots, m$ which is $O(m)$. Moreover the tail operator requires computing the sum of $(-1)^j \gamma_j \binom{m-1}{j}^{-1}$ for $j = 0, \dots, m-1$, where $\gamma_j = \sum_{i=0}^j (-1)^i \binom{m}{i} a_i$. Since $\gamma_{j+1} = \gamma_j + (-1)^{j+1} \binom{m}{j+1} a_{j+1}$, computing the head operator is $O(m)$.

Given the constants $\binom{i}{j}$ the operator $\text{Quot}_k(f, \sum_{j=0}^n b_j \beta_{j,n})$ is of complexity $O(k - (m - n))$, which is $O(k)$, and the operator $\text{Match}_k(f, \sum_{j=0}^n b_j \beta_{j,n})$ requires computing $a_j \binom{n}{j} \binom{m}{j}^{-1}$ for $j = 0, \dots, n$ which is $O(n)$. \square

Now we are ready to show the complexity of division.

Theorem 26 *The number of arithmetic operations required for division in the Bernstein basis is*

$$O(m^2 - \min(m - n, n - 1)^2).$$

Proof: We observe that $\text{Quot}_k(f, \sum_{j=0}^n b_j \beta_{j,n})$ requires $\binom{k-(m-n)}{j}$, for all $j = 0, \dots, k - (m - n)$ and that $\text{Match}_k(f, \sum_{j=0}^n b_j \beta_{j,n})$ requires the constants $\binom{n}{j}$ and $\binom{m}{j}$, for $j = 0, \dots, n$. Furthermore we observe that $\text{Head}_m(\sum_{j=0}^m b_j \beta_{j,m})$ and $\text{Tail}_m(\sum_{j=0}^m b_j \beta_{j,m})$ require the constants $\binom{m}{j}$, for $j = 0, \dots, m$ and $\binom{m-1}{j}$, for $j = 0, \dots, m-1$. Thus overall the division algorithm requires the constants $\binom{l}{j}$, for $j = 0, \dots, l$ and for $l = \min(m - n, n - 1), \dots, m$. Therefore Lemma 24 provides for all required constants.

Considering $n \leq m$, by Lemmas 24, 25 and Theorem 7 the division algorithm is of order $m^2 - \min(m - n, n - 1)^2 + (m + n)(m - n + 1)$. This is equivalent to $O(m^2 - \min(m - n, n - 1)^2 + m^2 - n^2)$ which is $O(m^2 - \min(m - n, n - 1)^2)$. \square

4.3 Correctness

We prove the correctness of the definitions of the basic operators in the previous section. The correctness follows from two theorems, 32 and 29, which respectively verify the defining properties of the head/tail and match/quot operators. Before proving these theorems we give some auxiliary lemmas.

Lemma 27

$$x^{j-1} (1 - x)^{m-(j-1)} = x^{j-1} (1 - x)^{(m-1)-(j-1)} - x^j (1 - x)^{m-j}.$$

Proof: $x^{j-1} (1 - x)^{m-(j-1)} = x^{j-1} (1 - x)^{(m-1)-(j-1)} \cdot 1 + x^{j-1} (1 - x)^{m-j} (-x)$. \square

Lemma 28

$$x^j (1 - x)^{m-j} = (-1)^{m-j} x^m + \sum_{i=j}^{m-1} (-1)^{i-j} x^i (1 - x)^{(m-1)-i}. \quad (3)$$

Proof: Proof by induction on $(m - j)$. For $m - j = 0$, that is, $j = m$, we have

$$x^m (1 - x)^0 = (-1)^0 x^m + 0.$$

Next we assume (3) and show (3) after replacing $m - j$ with $m - j + 1$, that is, after replacing j with $j - 1$. Thus we show

$$x^{j-1} (1 - x)^{m-(j-1)} = (-1)^{m-(j-1)} x^m + \sum_{i=j-1}^{m-1} (-1)^{i-(j-1)} x^i (1 - x)^{(m-1)-i}.$$

By Lemma 27 and by (3), we have

$$\begin{aligned} x^{j-1} (1 - x)^{m-(j-1)} &= x^{j-1} (1 - x)^{(m-1)-(j-1)} - x^j (1 - x)^{m-j} = \\ &= x^{j-1} (1 - x)^{(m-1)-(j-1)} - ((-1)^{m-j} x^m + \sum_{i=j}^{m-1} (-1)^{i-j} x^i (1 - x)^{(m-1)-i}). \end{aligned}$$

□

Theorem 29

$$\begin{aligned} \sum_{j=0}^m a_j \beta_{j,m} &= \left(\sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j \right) x^m \\ &\quad + \sum_{j=0}^{m-1} (-1)^j \left(\sum_{i=0}^j (-1)^i \binom{m}{i} a_i \right) \binom{m-1}{j}^{-1} \beta_{j,m-1}. \end{aligned}$$

Furthermore, $\sum_{j=0}^m (-1)^j \binom{m}{j} a_j$ is the coefficient of x^m in $\sum_{j=0}^m a_j \beta_{j,m}$.

Proof: By Lemma 28,

$$\begin{aligned} \sum_{j=0}^m a_j \beta_{j,m} &= \sum_{j=0}^m a_j \binom{m}{j} \left((-1)^{m-j} x^m + \sum_{i=j}^{m-1} (-1)^{i-j} x^i (1 - x)^{(m-1)-i} \right) \\ &= \sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j x^m + \sum_{j=0}^m a_j \binom{m}{j} \left(\sum_{i=j}^{m-1} (-1)^{i-j} x^i (1 - x)^{(m-1)-i} \right) \\ &= \sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j x^m + \sum_{j=0}^m \sum_{i=j}^{m-1} (-1)^{i-j} \binom{m}{j} a_j x^i (1 - x)^{(m-1)-i} \\ &= \sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j x^m + \sum_{j=0}^{m-1} \sum_{i=j}^{m-1} (-1)^{i-j} \binom{m}{j} a_j x^i (1 - x)^{(m-1)-i}. \end{aligned}$$

Next we change the order of summation in the last double sum. The summation indices of the double sum are the solutions of the set of inequalities

$$0 \leq j \leq m - 1, \quad 0 \leq i \leq m - 1, \quad j \leq i \leq m - 1.$$

This set of inequalities is equivalent to

$$0 \leq j \leq m - 1, \quad 0 \leq i \leq m - 1, \quad 0 \leq j \leq i.$$

Therefore $\sum_{j=0}^m a_j \beta_{j,m}$ equals

$$\begin{aligned} & \sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j x^m + \sum_{i=0}^{m-1} \sum_{j=0}^i (-1)^{i-j} \binom{m}{j} a_j x^i (1-x)^{(m-1)-i} \\ &= \sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j x^m + \sum_{i=0}^{m-1} (-1)^i \left(\sum_{j=0}^i (-1)^j \binom{m}{j} a_j \right) x^i (1-x)^{(m-1)-i} \\ &= \sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j x^m + \sum_{i=0}^{m-1} (-1)^i \left(\sum_{j=0}^i (-1)^j \binom{m}{j} a_j \right) \binom{m-1}{i}^{-1} \beta_{i,m-1}. \end{aligned}$$

Furthermore, we observe that $\sum_{j=0}^m (-1)^{m-j} \binom{m}{j} a_j$ is the coefficient of x^m in the polynomial $\sum_{j=0}^m a_j \beta_{j,m}$ because $\beta_{i,m-1}$ is of degree $m-1$. \square

Next we study the match and quot operators. We start with some auxiliary lemmas.

Lemma 30 $\beta_{j,m} = \binom{m}{j} \binom{n}{j}^{-1} (1-x)^{m-n} \beta_{j,n}$ if $m \geq n$.

Proof: $(1-x)^{m-j} = (1-x)^{m-n} (1-x)^{n-j}$. \square

Lemma 31 For $k \geq m-n$

$$(1-x)^{m-n} = \sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} x^j (1-x)^{k-j}.$$

Proof:

$$\begin{aligned} (1-x)^{m-n} &= (x + (1-x))^{k-(m-n)} (1-x)^{m-n} \\ &= \left(\sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} x^j (1-x)^{k-(m-n)-j} \right) (1-x)^{m-n} \\ &= \sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} x^j (1-x)^{k-j}. \end{aligned}$$

\square

The next theorem determines a multiple of a polynomial represented in Bernstein basis. This multiple is denoted by the operator Match and the factor by the operator Quot in the division algorithm.

Theorem 32 For $k \geq m - n$

$$\sum_{j=0}^n a_j \binom{n}{j} \binom{m}{j}^{-1} \beta_{j,m} = \left(\sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} \binom{k}{j}^{-1} \beta_{j,k} \right) \left(\sum_{j=0}^n a_j \beta_{j,n} \right).$$

Proof: By Lemmas 30 and 31,

$$\begin{aligned} \sum_{j=0}^n a_j \binom{n}{j} \binom{m}{j}^{-1} \beta_{j,m} &= \sum_{j=0}^n a_j (1-x)^{m-n} \beta_{j,n} \\ &= \left(\sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} x^j (1-x)^{k-j} \right) \left(\sum_{j=0}^n a_j \beta_{j,n} \right). \end{aligned}$$

□

5 Conclusion and Future Directions

The current work provided polynomial division algorithms over the Lagrange and Bernstein basis of quadratic worst-case complexity. Future work may include studying if the computational complexities of the provided algorithms can be improved further and extending the algorithms to multi-variate polynomial division.

For practical applications it will be of interest to study numerical properties of these algorithms because in practice computations are often performed with floating point numbers. The case of Bernstein basis is particularly interesting. That is, polynomials presented in Bernstein basis have been shown to be well suited for stable numerical computations [13]. However, [18] has shown that the classical polynomial division algorithm *in power basis* is numerically quite unstable in certain cases.

References

1. A. Amiraslani. Dividing polynomials when you only know their values. In L. Gonzalez-Vega and T. Recio, editors, *Proceedings of Encuentros de Álgebra Computacional y Aplicaciones (EACA) 2004*, pages 5–10, 2004. <http://www.orcca.on.ca/TechReports/2004/TR-04-01.html>.
2. A. Amiraslani. *New Algorithms for Matrices, Polynomials and Matrix Polynomials*. PhD thesis, University of Western Ontario, London, Ontario, Canada, 2006.
3. D. A. Aruliah, Robert M. Corless, Laureano Gonzalez-Vega, and Azar Shakoori. Geometric applications of the bezout matrix in the lagrange basis. In *SNC '07: Proceedings of the 2007 international workshop on Symbolic-numeric computation*, pages 55–64, New York, NY, USA, 2007. ACM.
4. D. A. Aruliah, Robert M. Corless, Azar Shakoori, Laureano Gonzalez-Vega, and Ignacio F. Rua. Computing the topology of a real algebraic plane curve whose equation is not directly available. In *SNC '07: Proceedings of the 2007 international workshop on Symbolic-numeric computation*, pages 46–54, New York, NY, USA, 2007. ACM.

5. S. Barnett. Division of generalized polynomials using the comrade matrix. *Linear Algebra Appl.*, 60:159–175, 1984.
6. S. Barnett. Euclidean remainders for generalized polynomials. *Linear Algebra Appl.*, 99:111–122, 1988.
7. Stephen Barnett. *Polynomials and linear control systems*, volume 77 of *Monographs and Textbooks in Pure and Applied Mathematics*. Marcel Dekker Inc., New York, 1983.
8. Dario A. Bini, Luca Gemignani, and Joab R. Winkler. Structured matrix methods for CAGD: an application to computing the resultant of polynomials in the Bernstein basis. *Numer. Linear Algebra Appl.*, 12(8):685–698, 2005.
9. Alin Bostan and Éric Schost. Polynomial evaluation and interpolation on special sets of points. *J. Complexity*, 21(4):420–446, 2005.
10. Howard Cheng and George Labahn. On computing polynomial GCDs in alternate bases. In *ISSAC 2006*, pages 47–54. ACM, New York, 2006.
11. R. Corless. Generalized companion matrices in the lagrange basis. In L. Gonzalez-Vega and T. Recio, editors, *Proceedings of Encuentros de Álgebra Computacional y Aplicaciones (EACA) 2004*, pages 317–322, 2004. <http://www.apmaths.uwo.ca/rcorless/frames/PAPERS/PABV/EACA2004Corless.pdf>.
12. Gema M. Diaz-Toca and Laureano Gonzalez-Vega. Barnett’s theorems about the greatest common divisor of several univariate polynomials through Bezout-like matrices. *J. Symbolic Comput.*, 34(1):59–81, 2002.
13. R. T. Farouki and T. N. T. Goodman. On the optimal stability of the Bernstein basis. *Math. Comp.*, 65(216):1553–1566, 1996.
14. L. Gemignani. Manipulating polynomials in generalized form. Technical Report TR-96-14, Università di Pisa, Dipartimento di Informatica, Corso Italia 40, 56125 Pisa, Italy, December 1996.
15. R. Goldman. *Pyramid Algorithms: A Dynamic Programming Approach to Curves and Surfaces for Geometric Modeling*. The Morgan Kaufmann Series in Computer Graphics. Morgan Kaufmann, 1st edition edition, July 2002.
16. Vaidyanath Mani and Robert E. Hartwig. Generalized polynomial bases and the Bezoutian. *Linear Algebra Appl.*, 251:293–320, 1997.
17. John Maroulas and Stephen Barnett. Greatest common divisor of generalized polynomial and polynomial matrices. *Linear Algebra Appl.*, 22:195–210, 1978.
18. Hans J. Stetter. *Numerical polynomial algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2004.
19. Y.-F. Tsai and R. T. Farouki. Algorithm 812: BPOLY: An object-oriented library of numerical algorithms for polynomials in Bernstein form. *ACM Transactions on Mathematical Software*, 27(2):267–296, 2001.
20. A. Vries-Baayens. *CAD product data exchange: conversions for curves and surfaces*. PhD thesis, Delft University, 1991.
21. F. Winkler. *Polynomial algorithms in computer algebra*. Texts and monographs in symbolic computation. Springer-Verlag, Vienna, 1996.
22. Joab R. Winkler. A resultant matrix for scaled Bernstein polynomials. *Linear Algebra Appl.*, 319(1-3):179–191, 2000.
23. Joab R. Winkler. Properties of the companion matrix resultant for Bernstein polynomials. In *Uncertainty in geometric computations*, volume 704 of *Kluwer Internat. Ser. Engrg. Comput. Sci.*, pages 185–198. Kluwer Acad. Publ., Boston, MA, 2002.
24. Joab R. Winkler. A companion matrix resultant for Bernstein polynomials. *Linear Algebra Appl.*, 362:153–175, 2003.