ESKİŞEHİR TEKNİK ÜNİVERSİTESİ BİLİM VE TEKNOLOJİ DERGİSİ B- TEORİK BİLİMLER

Eskişehir Technical University Journal of Science and Technology B- Theoretical Sciences

2023, 11(2), pp. 158-166, DOI: 10.20290/estubtdb.1279073

RESEARCH ARTICLE

HERMITE INTERPOLATION WITH DICKSON POLYNOMIALS AND BERNSTEIN BASIS POLYNOMIALS

Erdal İMAMOĞLU *

[1] Department of Mathematics, Faculty of Arts and Sciences, Kirklareli University, Kirklareli, Turkey

ABSTRACT

In this manuscript we introduce three new algorithms: (1) An algorithm to recover an unknown polynomial in terms of Dickson polynomials of the first kind, (2) an algorithm to recover an unknown polynomial in terms Dickson polynomials of the second kind, (3) an algorithm to recover an unknown polynomial in terms of Bernstein basis polynomials, from given black boxes for the polynomial itself and its first derivative. In each algorithm, we assume that the unknown polynomial has a sparse representation in the corresponding basis. The methods presented use transformations from Dickson polynomials to Laurent polynomials, a transformation from Bernstein basis polynomials to Laurent polynomials, and a recently developed algorithm as a middle step.

Keywords: Hermite Interpolation, Sparse Polynomials, Dickson Polynomials, Bernstein Basis Polynomials, Algorithms

1. INTRODUCTION

Hermite interpolation is a method of reconstructing an unknown polynomial $f(x)$ by using known evaluations of $f(x)$ and known evaluations of the first few derivatives of $f(x)$. More details about Hermite interpolation can be found at [1] and references therein. In this manuscript, we deal with sparse Hermite interpolation.

A sparse Hermite interpolation algorithm is presented in [2]: Let $f(x) = \sum_{j=1}^{t} c_j x^{e_j} \in K[x, x^{-1}]$ be an unknown sparse univariate Laurent polynomial, i.e. an element in $K[x, x^{-1}]$, in Laurent polynomial basis with $t \ll \deg(f)$ terms, where K is a field and its' characteristic is 0 or a prime p, and where for all $j, c_j \neq 0, e_1 < e_2 < \cdots < e_t$. Let $k \in K - \{0,1\}$. Let black boxes for $f(x)$ and $f'(x)$ be given. [2] introduces a procedure to rebuild the unknown polynomial $f(x)$ from the data sets $\{(k^s, f(k^s))\}_{s=0}^{m}$ and $\{(k^s, f'(k^s))\}_{s=0}^{m}$ where $m = t + \left\lceil \frac{t+1}{2} \right\rceil - 1$. Here the tuples $(*, f(*))$ and $(*, f'(*))$ can be computed with given black boxes. The algorithm presented in [2], which is based on Prony's sparse polynomial interpolation algorithm (a.k.a. Ben-or & Tiwari's Algorithm) [3,4], performs sparse Hermite interpolation using those $2m + 2$, where $t \ll \deg(f)$, data points above. The method in [2] uses "less data points" than the previously known Hermite interpolation algorithms use to reconstruct the unknown polynomial $f(x)$.

Remark 1.1 We note that a black box for an unknown polynomial $f(x)$ is a known mathematical object that takes a value k and evaluates $f(k)$ without revealing any information about the unknown polynomial $f(x)$. Here we assume a black box for a polynomial always computes the correct evaluation with no error. See [5] for more details about computations with black boxes.

*Corresponding Author: eimamoglu@klu.edu.tr
Received: 07.04.2023 Published: 28.08.2023

Any polynomial $f(x) \in K[x]$ can be represented in terms of Dickson polynomials (both the first and the second kind). A degree n polynomial with real coefficients can be represented in terms of degree n Bernstein basis polynomials. We want to replace the Laurent polynomial basis with Dickson polynomials and Bernstein basis polynomials and aim to develop new sparse Hermite interpolation algorithms that work directly with those bases.

In this text, we present three new algorithms that solve the following three problems. The algorithms in the present manuscript perform sparse Hermite interpolation with Dickson polynomials (both the first and the second kind) and Bernstein basis polynomials. The algorithms use transformations from Dickson polynomials to Laurent polynomials, a transformation from Bernstein basis polynomials to Laurent polynomials, and the algorithm given in [2] as a middle step.

Problem 1.1

i. Let

$$f(x) = \sum_{j=0}^{t} c_j D_{e_j}(x, a) \in K[x]$$

where for all $j, c_j \neq 0, e_1 < e_2 < \cdots < e_t$, be an unknown polynomial. Here $D_{e_j}(x, a)$ is the Dickson polynomial of the first kind of degree e_j. Here we assume that $t \ll \deg(f)$, i.e., $f(x)$ has sparse representation in terms of Dickson polynomials of the first kind.

Construct $f(x)$ from given black boxes for $f(x)$ and $f'(x)$, $a \in K$, the integer t, from the sets of tuples $\{(k^s, f(k^s))\}_{s=0}^{m}$ and $\{(k^s, f'(k^s))\}_{s=0}^{m}$ where $k \in K$ and $m = t + \left\lceil \frac{t+1}{2} \right\rceil - 1$. Here the data points are computed by the given black boxes.

ii. Let

$$f(x) = \sum_{j=0}^{t} c_j E_{e_j}(x, a) \in K[x]$$

where for all $j, c_j \neq 0, e_1 < e_2 < \cdots < e_t$, be an unknown polynomial. Here $E_{e_j}(x, a)$ is the Dickson polynomial of the second kind of degree e_j. Here we assume that $t \ll \deg(f)$, i.e., $f(x)$ has sparse representation in terms of Dickson polynomials of the second kind.

Construct $f(x)$ from given a black boxes for $f(x)$ and $f'(x)$, $a \in K$, the integer t, from the sets of tuples $\{(k^s, f(k^s))\}_{s=0}^{m}$ and $\{(k^s, f'(k^s))\}_{s=0}^{m}$ where $k \in K$ and $m = t + \left\lceil \frac{t+1}{2} \right\rceil - 1$. Here the data points are computed by the given black boxes.

iii. Let

$$f(x) = \sum_{j=0}^{t} c_j B_{e_j,n}(x) \in K[x]$$

where for all $j, c_j \neq 0, e_1 < e_2 < \cdots < e_t$, be an unknown polynomial. Here $B_{e_j,n}(x)$ is the e_j-th Bernstein basis polynomial of degree n. Here we assume $n = \deg(f(x))$ is known, $K = \mathbb{R}$, and $t \ll \deg(f)$, i.e., $f(x)$ has sparse representation in terms of Bernstein basis polynomials.

Construct $f(x)$ from given black boxes for $f(x)$ and $f'(x)$, $a \in K$, the integer t, $n = \deg(f(x))$ from the sets of tuples $\{(k^s, f(k^s))\}_{s=0}^{m}$ and $\{(k^s, f'(k^s))\}_{s=0}^{m}$ where $k \in K$ and $m = t + \left\lceil \frac{t+1}{2} \right\rceil - 1$. Here the data points are computed by the given black boxes.

Before we state our procedures, we briefly mention about Dickson polynomials and Bernstein basis polynomials.

1.1. Dickson Polynomials

Dickson polynomials are introduced in [6]. Let K be a finite field with characteristic p and $a \in K$. Degree n Dickson polynomial of the first kind, $D_n(x, a)$, can be defined by the following recursion:

$$D_0(x, a) \coloneqq 2$$
$$D_1(x, a) \coloneqq x$$
$$D_n(x, a) \coloneqq xD_{n-1}(x, a) - aD_{n-2}(x, a), \forall n \geq 2.$$

Similarly, degree n Dickson polynomial of the second kind, $E_n(x, a)$, can be defined by the same recursion as above, but with a different zero-degree polynomial:

$$E_0(x, a) \coloneqq 1$$
$$E_1(x, a) \coloneqq x$$
$$E_n(x, a) \coloneqq xE_{n-1}(x, a) - aE_{n-2}(x, a), \forall n \geq 2.$$

Dickson polynomials form a K vector space bases for $K[x]$: Any $f(x) \in K[x]$ can be represented in terms of Dickson polynomials (both the first and the second kind).

Dickson polynomials are one of the examples of many orthogonal polynomials and they occur in various areas of mathematical research, such as cryptography and number theory [8,9]. The polynomials possess many useful properties. Details of Dickson polynomials and their further properties can be found at [6-9] and references in [6-9].

1.2 Bernstein Basis Polynomials

The i-th degree n Bernstein basis polynomial, which is denoted by $B_{i,n}(x)$, is defined as

$$B_{i,n}(x) = \binom{n}{i} x^i (1-x)^{n-i}.$$

Here $\binom{n}{i}$ denotes the binomial coefficient. The set $\{B_{s,n}(x)\}_{s=0}^{n}$ form a vector space basis (a.k.a. Bernstein-Bezier basis) for the polynomials in Π_n, where Π_n is the vector space of polynomials of degree $\leq n$ with real coefficients. Bernstein-Bezier basis is the standard way of representing a polynomial curve. We refer to [10,11] for further properties of Bernstein basis polynomials.

2. DISCUSSION AND ALGORITHMS

2.1. Sparse Hermite Interpolation with Dickson Polynomials of the First Kind

In [9], it is stated that Dickson polynomials satisfy the transformation formulas below: If $x \neq 0$ and $x^2 \neq a$,

$$D_n\left(x+\frac{a}{x},a\right)=x^n+\left(\frac{a}{x}\right)^n$$

$$E_n\left(x+\frac{a}{x},a\right)=\frac{x^{n+1}-\left(\frac{a}{x}\right)^{n+1}}{x-\left(\frac{a}{x}\right)}.$$

If we let $b^2=a$, then:

$$D_n\left(bx+\frac{a}{bx},a\right)=b^nx^n+\left(\frac{a}{bx}\right)^n=b^n\left(x^n+\frac{1}{x^n}\right) \tag{1}$$

$$\left(bx-\frac{a}{bx}\right)E_n\left(bx+\frac{a}{bx},a\right)=\left(bx-\frac{a}{bx}\right)\left(\frac{b^{n+1}x^{n+1}-\left(\frac{a}{bx}\right)^{n+1}}{bx-\left(\frac{a}{bx}\right)}\right)=b^{n+1}\left(x^{n+1}-\frac{1}{x^{n+1}}\right). \tag{2}$$

Equations (1) and (2) are also used in [12] to perform sparse polynomial interpolation in Dickson polynomial bases.

Assume that $f(x)=\sum_{j=0}^{t}c_jD_{e_j}(x,a)$. Then, with the help of Equation (1), we can define $g(x)$ from $f(x)$:

$$\begin{aligned} g(x) &\coloneqq f\left(b\left(x+\frac{1}{x}\right)\right)\\ &=f\left(bx+\frac{a}{bx}\right)\\ &=\sum_{j=0}^{t}c_jD_{e_j}\left(bx+\frac{a}{bx},a\right)\\ &=\sum_{j=1}^{t}G_j\left(x^{e_j}+\frac{1}{x^{e_j}}\right)\in K[x,x^{-1}] \end{aligned} \tag{3}$$

where $G_j=b^{e_j}c_j$.

Here, $g(x)$ has $T=2t$ terms in Laurent polynomial bases and $g\left(k^i\right)=g\left(k^{-i}\right)$ for $k\in K$. To compute the two evaluations $g\left(k^i\right)$ and $g\left(k^{-i}\right)$, we need to evaluate $f(x)$ only once at $x=b\left(k^i+\frac{1}{k^i}\right)$.

Here, we have

$$g'(x)=f'\left(b\left(x+\frac{1}{x}\right)\right)\left(b\left(1-\frac{1}{x^2}\right)\right)$$

and

$$g'(x)=\sum_{j=1}^{t}\frac{G_je_j}{x}\left(x^{e_j}-\frac{1}{x^{e_j}}\right).$$

Note that, $k^i g'(k^i) = -(k^{-i} g'(k^{-i}))$. To compute the two evaluations $g'(k^i)$ and $g'(k^{-i})$), we need to evaluate $f'(x)$ only once at $x = b\left(k^i + \frac{1}{k^i}\right)$.

We make use of Equation (3) to present Algorithm 2.1.1 below that solves Problem 1.1.i. Algorithm 2.1.1 first uses Equation (3) to convert Problem 1.1.i to another problem that the Algorithm in [2] can solve, then uses Algorithm [2], and then recovers the coefficient-degree tuples (c_j, e_j) such that $f(x) = \sum_{j=0}^{t} c_j D_{e_j}(x, a)$.

Algorithm 2.1.1

Input:

- Black boxes for $f(x)$ and $f'(x)$.
- The integer t.
- $k \in K - \{0,1\}$.
- $a \in K - \{0\}$ such that $b^2 = a$.

Output:

- The c_j and the e_j such that $f(x) = \sum_{j=0}^{t} c_j D_{e_j}(x, a)$.
- The δ_j and the ε_j such that for $f_\varepsilon(x) = \sum_{j=0}^{t} \delta_j D_{\varepsilon_j}(x, a)$; $f\left(k^{i_0+i}\right) = f_\varepsilon\left(k^{i_0+i}\right)$ and $f'\left(k^{i_0+i}\right) = f'_\varepsilon\left(k^{i_0+i}\right)$.

1.

i. Use Equation (3) and form $g(x)$.

ii. Let $\ell = -\left\lceil \frac{3t-1}{2} \right\rceil$.

a. By using the black box for $f(x)$, for $i = 0,1, \ldots, |\ell|, \ldots 3t-1$, compute the $a_i = g\left(k^{\ell+i}\right)$ by using $g(x) = f\left(b\left(x + \frac{1}{x}\right)\right)$.

Use the equality $g\left(k^{\ell+i}\right) = g\left(k^{-\ell-i}\right)$ to generate the a_i with less computation.

b. By using the black box for $f'(x)$, for $i = 0,1, \ldots, |\ell|, \ldots, 3t-1$, compute the $a'_i = g'\left(k^{\ell+i}\right)$ by using $g'(x) = f'\left(b\left(x + \frac{1}{x}\right)\right)\left(b\left(1 - \frac{1}{x^2}\right)\right)$.

Use the equality $k^{\ell+i} g'\left(k^{\ell+i}\right) = -\left(k^{-\ell-i} g'\left(k^{-\ell-i}\right)\right)$ to generate the a'_i with less computation.

We encounter the same scenario as in Section 5.2 of [1]: Similarly, here we have $i_0 = \ell = -\left\lceil \frac{3t-1}{2} \right\rceil$, $g\left(k^{\ell+i}\right) = g\left(k^{-\ell-i}\right)$, $k^{\ell+i} g'\left(k^{\ell+i}\right) = -\left(k^{-\ell-i} g'\left(k^{-\ell-i}\right)\right)$. As stated in Section 5.2 of [1], we can compute the a_i and the a'_i above from $2(|\ell| + 1) \leq 3t + 2$ values of $f(x)$ and $f'(x)$. To generate those values of $f(x)$ and $f'(x)$, we need to use given black boxes for $f(x)$ and $f'(x)$ only $\leq t + \left\lfloor \frac{t}{2} \right\rfloor + 1$ times.

2. Use the Algorithm 2.1 in [1] with inputs $T = 2t, k, i_0 = \ell, r = \left\lfloor \frac{t}{2} \right\rfloor$, and $a_i = g\left(k^{\ell+i}\right)$ and $a_i'' = k^{\ell+i}a_i' = k^{\ell+i}g'\left(k^{\ell+i}\right)$. Note that here we have $3T = 6t$ values of $g(x)$ and $g'(x)$.

3.

i. If Step 2 decides there is no T sparse polynomial $g(x)$ in Laurent basis that interpolates a_i and a_i'', then print that information and stop.

ii. If $char(K) = 0$, or, $char(K) > 0$ and $k^s \neq 1$ for all $s \geq 1$, then the algorithm in Step 2 returns the G_j and the e_j such that $g(x) = \sum_{j=1}^{t} G_j \left(x^{e_j} + \frac{1}{x^{e_j}}\right)$.

In this case, compute the c_j from $G_j = b^{e_j}c_j$ and return the c_j and the e_j such that $f(x) = \sum_{j=0}^{t} c_j D_{e_j}(x, a)$.

iii. If $char(K) > 0$ and there exists $s \geq 2$ such that $k^s = 1$, then the algorithm in Step 2 returns Γ_j and ε_j such that $g_\varepsilon(x) = \sum_{j=1}^{t} \Gamma_j \left(x^{\varepsilon_j} + \frac{1}{x^{\varepsilon_j}}\right)$ such that $g\left(k^{\ell+i}\right) = g_\varepsilon\left(k^{\ell+i}\right)$ and $g'\left(k^{\ell+i}\right) = g_\varepsilon'\left(k^{\ell+i}\right)$.

In this case, compute the δ_j from $\Gamma_j = b^{\varepsilon_j}c_j$ and return the c_j and the ε_j such that $f_\varepsilon(x) = \sum_{j=0}^{t} \delta_j D_{\varepsilon_j}(x, a)$, such that $f\left(k^{\ell+i}\right) = f_\varepsilon\left(k^{\ell+i}\right)$ and $f'\left(k^{\ell+i}\right) = f_\varepsilon'\left(k^{\ell+i}\right)$.

2.2 Sparse Hermite Interpolation with Dickson Polynomials of the Second Kind

Assume that $f(x) = \sum_{j=0}^{t} c_j E_{e_j}(x, a)$. Then, with the help of Equation (2), we can define $h(x)$ from $f(x)$:

$$\begin{aligned} h(x) &:= \left(b\left(x - \frac{1}{x}\right)\right) f\left(b\left(x + \frac{1}{x}\right)\right) \\ &= \left(bx - \frac{a}{bx}\right) f\left(bx + \frac{a}{bx}\right) \\ &= \left(bx - \frac{a}{bx}\right) \sum_{j=0}^{t} c_j E_{e_j}\left(bx + \frac{a}{bx}, a\right) \\ &= \sum_{j=1}^{t} H_j \left(x^{e_j+1} - \frac{1}{x^{e_j+1}}\right) \in K[x, x^{-1}] \end{aligned} \tag{4}$$

where $H_j = b^{e_j+1}c_j$.

Here, $h(x)$ has $T = 2t$ terms in Laurent polynomial bases and $h\left(k^i\right) = -h\left(k^{-i}\right)$ for $k \in K$. To compute two evaluations $h\left(k^i\right)$ and $h\left(k^{-i}\right)$, we need to evaluate $f(x)$ only once at $x = b\left(k^i + \frac{1}{k^i}\right)$.

Here, we have

$$h'(x) = b\left(1 + \frac{1}{x^2}\right) f\left(b\left(x + \frac{1}{x}\right)\right) + a\left(x - \frac{1}{x}\right)\left(1 - \frac{1}{x^2}\right) f'\left(b\left(x + \frac{1}{x}\right)\right)$$

and

$$h'(x) = \sum_{j=1}^{t} \frac{H_j(e_j+1)}{x}\left(x^{e_j+1} + \frac{1}{x^{e_j+1}}\right).$$

Note that, $k^i h'(k^i) = k^{-i}h'(k^{-i})$. To compute the two evaluations $h'(k^i)$ and $h'(k^{-i})$, we need to evaluate $f(x)$ and $f'(x)$ only once at $x = b\left(k^i + \frac{1}{k^i}\right)$.

One can make use of Equation (4) and can design an algorithm (which is similar to Algorithm 2.1.1) that solves Problem 1.1.ii.

2.3 Sparse Hermite Interpolation with Bernstein Basis Polynomials

In [13], it is introduced that

$$(1+x)^n B_{i,n}\left(\frac{x}{1+x}\right) = \binom{n}{i} x^i. \tag{5}$$

Assume $f(x) = \sum_{j=0}^{t} c_j B_{e_j,n}(x)$. Then, with the help of Equation (5), we can define $z(x)$ from $f(x)$:

$$\begin{aligned} z(x) &\coloneqq (1+x)^n f\left(\frac{x}{1+x}\right) \\ &= (1+x)^n \sum_{j=0}^{t} c_j B_{e_j,n}\left(\frac{x}{1+x}\right) \\ &= \sum_{j=0}^{t} Z_j\, x^{e_j} \end{aligned} \tag{6}$$

where $Z_j = \binom{n}{j} c_j$. Here $z(x)$ and $f(x)$ have the same number of terms and $z(k^i), z'(k^i)$ can be computed from $f\left(\frac{k^i}{1+k^i}\right), f'\left(\frac{k^i}{1+k^i}\right)$.

We can make use of Equation (6) and can design an algorithm that solves Problem 1.1.iii. Algorithm 2.3.1 first uses Equation (6) to convert Problem 1.1.iii to another problem that the Algorithm in [2] can solve, then uses Algorithm [2], and then recovers the coefficient-degree tuples (c_j, e_j) such that $f(x) = \sum_{j=0}^{t} c_j B_{e_j,n}(x)$.

Algorithm 2.3.1

Input:

- Black boxes for $f(x)$ and $f'(x)$.
- The integer t.
- An integer r such that $1 \le r \le t-1$.
- An integer ℓ.
- $k \in \mathbb{R} - \{0,1\}$.
- $n = \deg(f(x))$.

Output:

- The c_j and the e_j such that $f(x) = \sum_{j=0}^{t} c_j B_{e_j,n}(x)$.

1.
   i. Use Equation (6) and form $z(x)$.

   ii. By using black the boxes for $f(x)$ and $f'(x)$, for $i = 0, \ldots, 2t - r - 1$, compute $a_i = z(k^{\ell+i})$ and $a_i'' = k^{\ell+i} z'(k^{\ell+i})$ by using $z(x) = (1+x)^n f\left(\frac{x}{1+x}\right)$.

2. Use the algorithm 2.1 in [1] with inputs $t, k, i_0 = \ell, r$, and $a_i = z(k^{\ell+i})$ and $a_i'' = k^{\ell+i} z'(k^{\ell+i})$.

3.
   i. If Step 2 decides there is no t sparse polynomial $z(x)$ in Laurent basis that interpolates a_i and a_i'', then print that information and stop.

   ii. If Step 2 returns the Z_j and the e_j such that $z(x) = \sum_{j=0}^{t} Z_j\, x^{e_j}$, then compute the c_j from $Z_j = \binom{n}{j} c_j$, and then return the c_j and the e_j.

3. CONCLUSION

In this manuscript, we present three sparse Hermite interpolation algorithms: An algorithm that computes an unknown polynomial directly as a linear combination of Dickson polynomials of the first kind, an algorithm that recovers an unknown polynomial directly in terms of Dickson polynomials of the second kind, and an algorithm that rebuilds an unknown polynomial as a combination of Bernstein basis polynomials. Future work may include developing sparse Hermite interpolation algorithms that perform similar compuations with orthogonal polynomial bases, such as Legendre polynomials and Jacobi polynomials.

CONFLICT OF INTEREST

The author stated that there are no conflicts of interest regarding the publication of this article.

REFERENCES

[1] Von Zur Gathen J, Gerhard J. Modern computer algebra. Cambridge university press, 2013.

[2] Kaltofen EL. Sparse polynomial Hermite interpolation. In: ISSAC 2022 The International Symposium on Symbolic and Algebraic Computation Conference; 4-7 July 2022; Lille, France: ACM ISSAC'22, 469-478.

[3] Prony R. Essai experimental et analytique: sur les lois de la dilatabilite des fluides elastique et sur celles de la force expansive de la vapeur de l'eau et de la vapeur de l'alkool, a differentes temperatures. J. de l'Ecole Polytechnique, 1795; (1):24-76.

[4] Ben-Or M, Tiwari P. A deterministic algorithm for sparse multivariate polynomial interpolation. In: The twentieth annual ACM Symposium on Theory of Computing; 1988 New York, USA; 301-309.

[5] Kaltofen E, Trager B.M. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. Journal of Symbolic Computation 1990; 9(3), 301-320.

[6] Dickson LE. The analytic representation of substitutions on a power of a prime number of letters with a discussion of the linear group. Ann Math 1896; 1; 6, 65-120.

[7] Lindl R. Theory and applications of Dickson polynomials. World Scientific, 1991.

[8] Lindl R, Niederreiter H. Finite fields. Cambridge University Press, 1997.

[9] Mullen GL, Panario D. Handbook of finite fields. CRC Press. 2013.

[10] Farouki RT. The Bernstein polynomial basis: A centennial retrospective. Computer Aided Geometric Design 2012; 29(6), 379-419.

[11] Farin G. Curves and surfaces for computer aided geometric design. Academic Press, 1993.

[12] Imamoglu E, Kaltofen EL. A note on sparse polynomial interpolation in Dickson polynomial basis. ACM Communications in Computer Algebra 2020; 54(4), 125-128.

[13] Imamoglu E. Sparse polynomial interpolation with Bernstein polynomials. Turkish Journal of Mathematics 2021; 45(5), 2103-2107.