Computing Dimensionally Parametrized Determinant Formulas

Diplomarbeit
vorgelegt von
Mark Ziegelmann

nach einem Thema von Prof. Dr. Raimund Seidel
im Fachbereich 14, Informatik, der Universität des Saarlandes

Hiermit erkläre ich an Eides Statt, daß ich diese Diplomarbeit selbständig verfaßt und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Ferner habe ich die Arbeit noch keinem anderen Prüfungsamt vorgelegt.

Saarbrücken, im Dezember 1997

Danksagung

An dieser Stelle möchte ich mich bei Herrn Prof. Dr. Raimund Seidel für die Vergabe des interessanten Themas und für wertvolle Anregungen bedanken.

Des weiteren danke ich Markus Denny, Nicola Geismann und Frank Lehmann für viele fruchtbare Diskussionen. Inbesondere danke ich Frank Lehmann ebenfalls für das kritische Korrekturlesen der Arbeit.

Vielen Dank auch an Jochen Comes, der mich von den Fähigkeiten von LyX überzeugt hat und mir als Experte stets zur Seite stand. Bei allen Angehörigen des Lehrstuhls bedanke ich mich für die freundschaftliche Atmosphäre.

Mein besonderer Dank gilt meiner Familie und meiner Freundin, ohne deren Unterstützung und Verständnis ich mich nicht in dieser Weise meinem Studium hätte widmen können.

Contents

Introduction

Determinants have a long history in mathematics and arise in numerous applications. Consequently, they have been researched extensively which has yielded efficient algorithms for determinant computation of different matrix classes. Here we are not interested in the *value* of a determinant of fixed integer order but rather in the determinant *formula* of a specially structured matrix of symbolic dimension n. It is assumed that a certain simple structure of a matrix yields a corresponding special structure for its determinant formula. This problem has been investigated very early, Muir's treatise on the theory of determinants [Met60] contains a large number of early papers related to this subject motivated by applications in algebra and analysis or simply by the interesting structures of the matrix coefficients. Nowadays, many of these results tend to be forgotten or are buried in induction exercises in linear algebra books. Applications of dimensionally parametrized determinant formulas occur for example in the study of arbitrary dimensional geometric predicates in determinant form: If we want to prove a general statement for a special configuration then we need the determinant formula of the predicate.

The goal of this work is the investigation of determinant formulas for a number of important classes of specially structured matrices and the design of a software package for the computer algebra system Maple that tries to derive determinant formulas for specified matrices of these classes.

This work is organized as follows:

The first chapter is an introduction to determinants. We give a brief historical overview about the development of determinants, followed by the definition and the basic theoretical background of determinants. After an outline of possible applications we discuss general algorithms for determinant computation and motivate the desire to derive determinant formulas for specially structured matrices.

In Chapter 2 we derive general determinant formulas for matrix classes that only have bordering rows and columns as well as the main diagonal nonzero. Determinants of this kind arise for example in the formulation of special geometric predicates and will be important for the treatment of other matrix classes. We try to generalize the concept allowing another neighbouring nonzero diagonal or allowing the nonzero rows and columns to have arbitrary position. The chapter closes with a description of the implementation of a Maple package dealing with such matrices.

Chapter 3 is devoted to alternants. We present early results showing that an alternant is the product of a symmetric function with the difference product of its variables. With the help of these results we are able to derive determinant formulas for a restricted class of alternants. In the second part of the chapter we try to generalize the results for double alternants. The last part of the chapter focuses on the implementation of a Maple package that can handle a special class of alternants.

Chapter 4 discusses determinants of tridiagonal matrices and Hessenberg matrices that often arise in eigenvalue problems. We derive recurrence formulas for both classes and explicit formulas for a few special cases. The connection of continuants to continued fractions and Fibonacci numbers is pointed out. Implementation issues of a corresponding Maple package are covered in the last part of the chapter.

Chapter 5 deals with symmetric determinants. We distinguish different kinds of symmetries and try to obtain determinant formulas for special classes of these matrices making use of the results of Chapter 2 and Chapter 4. The implementation of a package dealing with some symmetric determinants is addressed towards the end of the chapter.

The closing chapter is a discussion of the theoretical and practical results and points out ideas for possible future work.

The appendix contains a brief description of the Maple system and the complete tutorial for the implemented Maple packages.

The packages for Maple V Release 4 including documentation can be downloaded at:

`http://www.cs.uni-sb.de/users/mdenny/Bewohner/mark/Determinant.html`

Chapter 1

Introduction to determinants

This chapter is an introduction to determinants. Starting with a short historical overview following [Bri83], we will then give a formal definition of determinants and cover the required theoretical framework. After a discussion of possible applications and general algorithms, we will motivate the desire to find determinant formulas for some specially structured matrices.

1.1 A brief history of determinants

Before determinants were systematically defined and mathematically investigated, they had appeared as tools to solve linear systems of equations. The Japanese mathematician Seki Shinsuke Kova, following traditional Chinese methods for numerically solving systems of linear equations, developed expressions that corresponded to the use of determinants. Independently, determinant–like expressions arose in European mathematics a little later, when Leibniz in a letter to L'Hospital (1683) gave a condition, in which circumstances a homogeneous system of three equations in three indeterminates has a non–trivial solution. His letter remained unpublished until the middle of the last century and therefore without influence on the development of determinants.

An immediately influencing approach on the use of determinants as computational expressions was Cramer's method for solving linear systems of equations (1750) which he developed researching plain algebraic curves. He gave a verbal description of "Cramer's rule", where determinants of the coefficients of the system implicitly appeared as nominator and denominator terms. However, Cramer did not define or investigate the arising polynomial functions that were later called determinants.

This defining step was due to Vandermonde in 1771 in a paper on linear elimination theory. He defined determinant functions of $n \times n$ (doubly indexed) variables through a recursive process, already discovered by Bezout while solving systems of linear equations. He also introduced his own notation for these functions. The order of the variables in a matrix scheme was arbitrary. Vandermonde formulated Cramer's rule using these functions and gave a proof (what Cramer had not done). Moreover, he discovered some properties of these functions: alternating sign when exchanging the indices corresponding to row or column exchanges; equality to zero when two rows or columns are identical; decomposition into subdeterminants.

Laplace (1772), Lagrange (1773) and Gauß (1801) continued the study of determinants and discovered other properties and applications. Laplace found the well known method of expanding a determinant. Lagrange used determinant expressions in number theory and in the computation of the volume of a pyramid in Euclidean space. Gauß used determinants in the investigation of quadratic forms in number theory. In the transformation of a quadratic form, Lagrange (for $n = 2$) and Gauß (for $n = 3$) observed special cases of the determinant multiplication theorem $\det({}^tS \cdot A \cdot S) = \det(A) \cdot \det(S)^2$.

This was the background of a general systematic study of Cauchy in 1812. He viewed determinants as functions of a quadratic scheme of variables ("système symétrique") and hence related them closely to matrices, even when not yet discussing them as linear mappings. In a generalization of the result of Gauß, he formulated the determinant multiplication theorem and introduced the composition of two $n \times n$ matrices (without explaining its significance). Cauchy also found several theorems about relations between subdeterminants. His systematic analysis opened the way to an interpretation of determinants which was dominant during the entire 19th century.

Grassmann, in his "Ausdehnungslehre" (1844), discovered a possibility to introduce determinants as n–fold "outer products" of vectors, which he studied in his work. His opinions on the geometry of vector spaces remained more or less unnoticed during the last century. Like the term of a vectorspace became a fundamental term in mathematics not until the second and third decade of this century, it was not until the twenties that a huge interest for a more abstract view of determinants, adjusted to the vector space term, arose. Consequently, for example Schreier and Sperner in their book on analytic geometry and algebra (1931) gave an axiomatic definition of a determinant for an n dimensional vector space V as a multilinear alternating mapping $V^n \to \mathbb{R}$. Thus, in modern mathematics, the explicit, systematic study of determinants started by Vandermonde and Cauchy was enlarged with a notational structural aspect. It is also interesting to note that the theory of determinants is much older than the theory of matrices.

1.2 Definition and basic properties of determinants

In this section we will cover the theoretical background of determinants which is relevant for the following chapters. Since the reader will be familiar with most of the concepts, we will refrain from a detailed presentation and give it a more enumerative character following [Jän91]. A detailed formal introduction including the basic algebraic terms and the term "matrix" which we assume to be known, can be found in any textbook on linear algebra (see for example [Jän91] [Bri83] [SW89]).

Consider quadratic matrices over a ring R, which will be denoted as $M(n \times n, R)$. Typically, R will be a field like $\mathbb{Q}$ or a ring of multivariate polynomials. We will define the determinant as a function mapping matrices over this ring into elements of the ring.

Theorem 1 (and Definition) *There is a unique mapping $|\cdot| : M(n \times n, R) \to R$ with the following properties:*

1. *$|\cdot|$ is linear with respect to each row.*
2. *If the (row) dimension is smaller than n then $|A| = 0$.*
3. *$|I| = 1$.*

The mapping $|\cdot|$ is called determinant *and $|A| \in R$ is called the* determinant *of A.*

Proof.

[Jän91] gives a proof of existence and uniqueness.

$\square$

Since we often refer to special diagonals of a matrix or its determinant, we will agree on some naming conventions.

Definition 1 *Let $A \in M(n \times n, R)$. We will call the elements a_{ii} for $i = 1, \dots, n$ the* main diagonal *of A, the elements $a_{i,i+1}$ for $i = 1, \dots, n-1$ the* first upper side diagonal *and the elements $a_{i+1,i}$ for $i = 1, \dots, n-1$ the* first lower side diagonal.

When we speak of the diagonal at position k, *we mean the elements $a_{i,i+k}$ for $0 \le k \le n-1$ or the elements $a_{i+k,i}$ for $-(n-1) \le k < 0$ respectively for $i = 1, \dots, n - |k|$.*

The elements $a_{i,n-i+1}$ for $i = 1, \dots, n$ will be called the counter main diagonal *of A.*

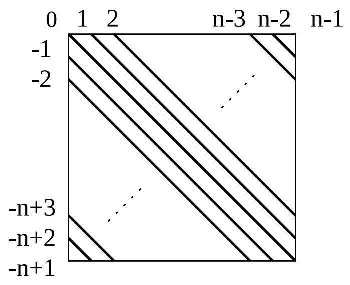

Figure 1: Diagonal positions

We will now present important properties of the determinant that will be used in almost all applications.

Lemma 1 *Let $|\cdot| : M(n \times n, R) \to R$ be a mapping with the properties 1. and 2. from above, then the following holds:*

1. *If we swap two rows of a matrix A, getting a matrix A', then $|A| = -|A'|$.*
2. *If we change the matrix A to a matrix A' by multiplying a scalar $\lambda \in R$ to one of its rows, then $|A'| = \lambda \cdot |A|$.*
3. *If we add a multiple of one row to another row of A, obtaining A', then $|A| = |A'|$.*

Definition 2 *Let $A \in M(n \times n, R)$. The determinant of order $n-1$ obtained from A by deleting the ith row and the jth column is called a* minor *of A and is denoted by $|A_{ij}|$. We will denote a minor of order r, obtained by deleting rows $i_1, \dots, i_r$ and columns $j_1, \dots, j_r$ by $|A_{i_1 i_2 \cdots i_r, j_1 j_2 \cdots j_r}|$.*

After knowing the definition and the basic properties of determinants, we are interested in actually computing their value. Laplace's famous minor expansion theorem provides an important method to recursively compute the value of a determinant.

Theorem 2 (Minor expansion) *If $A \in M(n \times n, R)$ then*

$$|A| = \sum_{i=1}^{n} (-1)^{i+j} a_{ij} |A_{ij}|.$$

Proof.

See for example [Jän91] for a proof that the so defined mapping $|\cdot|$ indeed has the properties 1. - 3. of Theorem 1.

$\square$

The previous theorem straightforwardly leads to the following lemma.

Lemma 2 *If $A \in M(n \times n, R)$ is an upper triangular matrix (i.e. $a_{ij} = 0$ for $i > j$), then the determinant is the product of the main diagonal elements:*

$$|A| = a_{11} a_{22} \cdots a_{nn}.$$

If ${}^tA$ denotes the transposed matrix of a matrix $A = (a_{ij})$, i.e. ${}^tA = (a_{ji})$, then we have the following result.

Lemma 3 *The identity $|A| = |{}^tA|$ holds for all $A \in M(n \times n, R)$.*

Proof.

Since the rank of rows and columns is equal, we have $\mathrm{rk}A = \mathrm{rk}^t A$. Obviously, the identity is true for the identity matrix I, thus it remains to prove that $|\cdot| : M(n \times n, R) \to R$ is linear in the columns too. The linearity of $|\cdot|$ in the jth column follows immediately from the column expansion formula $|A| = \sum_i (-1)^{i+j} a_{ij} |A_{ij}|$, since $|A_{ij}|$ is independent of the jth column of A (which is deleted).

$\square$

A consequence of this lemma is that we obtain a corresponding formula for the determinant expansion of the ith row of a matrix:

$$|A| = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} |A_{ij}|.$$

We will need another result concerning the determinant of a product of two matrices.

Lemma 4 *If $A, B \in M(n \times n, R)$ then*

$$|A \cdot B| = |A| \cdot |B|.$$

Closing, we want to take a look at the formula of Leibniz for the determinant of a matrix $A \in M(n \times n, R)$ which gives some obvious insight about the complexity of the resulting expression.

Theorem 3 (Leibniz) *Let $A \in M(n \times n, R)$ and Π_n be the set of all possible permutations of the numbers from 1 to n, i.e. the set of bijective mappings $\pi : \{1, \dots, n\} \to \{1, \dots, n\}$. The sign of a permutation is defined to be 1 if it is result of an even number of exchanges of neighbouring elements and defined to be -1 if this number is odd.*

The following formula holds:

$$|A| = \sum_{\pi \in \Pi_n} \mathit{sign}(\pi) \cdot a_{1\pi(1)} a_{2\pi(2)} \cdots a_{n\pi(n)}.$$

Proof.

Proving Leibniz' formula without using the term of a determinant, we have given yet another proof of the existence statement in Theorem 1. We only have to show that the mapping $M(n \times n, R) \to R$ defined by the right hand side of the formula has the properties 1. - 3. of Theorem 1.

1. All of the summands are linear in the rows, thus also the sum.

2. We have to show that the right hand side of the Leibniz formula vanishes if the row rank of A is less than n, which amounts to proving this for a matrix A with two equal rows. Let row i and row j be equal. If σ is the transposition that swaps i and j, and Π_{even} the set of permutations having sign 1 then we can write the right hand side of the Leibniz formula as

$$\sum_{\pi \in \Pi_{even}} (\mathrm{sign}(\pi) \cdot a_{1\pi(1)} \cdots a_{n\pi(n)} + \mathrm{sign}(\pi \circ \sigma) a_{1\pi\sigma(1)} \cdots a_{n\pi\sigma(n)}).$$

   Now, $a_{1\pi\sigma(1)} \cdots a_{n\pi\sigma(n)}$ results from $a_{1\pi(1)} \cdots a_{n\pi(n)}$ by replacing the ith factor $a_{i\pi(i)}$ by $a_{i\pi(j)}$ and $a_{j\pi(j)}$ by $a_{j\pi(i)}$. Since the rows i and j are equal, this changes nothing and the claim follows with $\mathrm{sign}(\pi \circ \sigma) = -\mathrm{sign}(\pi)$.

3. If A is the identity matrix then we only have one nonzero summand resulting from the identity permutation, thus $|I| = 1$.

$\square$

Note that there are $n!$ different permutation of n numbers, thus there are $n!$ products of n elements in this formula, implying that this formula is not very practical for large n.

1.3 Applications

Determinants arise in all different application areas. We will discuss a number of possible applications in the following.

1.3.1 Matrix inversion and systems of linear equations

Matrix inversion Every quadratic matrix A over a field K with $|A| \neq 0$ can be inverted (such matrices are called *non–singular*), i.e. there is a matrix A^{-1} such that $A \cdot A^{-1} = I$. How can we find this inverse ?

We define the complementary matrix $\widetilde{A}$ of A by

$$\widetilde{a_{ij}} = (-1)^{i+j}|A_{ji}|.$$

Now we are able to express the inverse A^{-1} of a non–singular matrix as

$$A^{-1} = \frac{1}{|A|} \cdot \widetilde{A}.$$

See for example [Jän91] for a proof.

Observe that every entry of the complementary matrix is a $(n-1) \times (n-1)$ determinant, hence we would have to compute $n^2 + 1$ determinants to obtain the inverse A^{-1} which is too inefficient. We will mention a more efficient method below.

Systems of linear equations One of the most important standard problems is the computation of the solution $x \in K^n$ of a linear equation system $Ax = b$ with $A \in M(n \times n, K)$ and $b \in K^n$. Cramer gave an explicit formula involving determinants for the solution of a linear system in 1750.

Cramer's Rule: Let $x, b \in K$ and $A \in M(n \times n, K)$ with $|A| \neq 0$. If $Ax = b$ then

$$x_i = \frac{\begin{vmatrix} a_{11} & \cdots & a_{1i-1} & b_1 & a_{1i+1} & \cdots & a_{1n} \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ a_{n1} & \cdots & a_{ni-1} & b_n & a_{ni+1} & \cdots & a_{nn} \end{vmatrix}}{|A|}, \quad \text{for } i = 1, \ldots, n.$$

However, this is not a practical method. The standard algorithm for solving a linear equation system is Gaussian elimination (see [GvL96] [Sto94] for details). The main idea is to convert a given system with elementary row operations into a triangular system which is easy to solve. It is also possible to compute the inverse of a matrix applying this method (See [Sto94] for details of the Gauß–Jordan algorithm). We will meet Gauß' fundamental method again later because it is also the most widely used algorithm for determinant computation.

1.3.2 Eigenvalue problems

In many problems in engineering or physics, it is necessary to determine $\lambda \in \mathbb{C}$ for $A \in M(n \times n, \mathbb{R})$ such that the homogeneous linear equation system

$$(A - \lambda I)x = 0$$

has a non–trivial solution $x \neq 0$.

A number $\lambda \in \mathbb{C}$ is called an *eigenvalue* of the matrix A, if there is a vector $x \neq 0$ with $Ax = \lambda x$. The corresponding vector x is called the *eigenvector* of A for the eigenvalue λ.

It is easy to see that λ is an eigenvalue of A if and only if

$$|A - \lambda I| = 0.$$

Note that $p(\mu) = |A - \mu I|$ is a polynomial in μ of degree n. The roots of p are exactly the eigenvalues of A. Hence the eigenvalue problem can be reduced to the problem of finding roots of a determinant polynomial.

It is important to note, however, that there are more efficient methods than computing the determinant polynomial $p(\mu)$ and its roots to find the eigenvalues (see [SB90] or [GvL96]).

1.3.3 Volume computation

For finite dimensional real affine spaces like $\mathbb{R}^n$ there is a close relation between determinants (and determinant functions) and volume computation (see [SW89] for details).

For example consider the affine vector space $\mathbb{R}^n$ with basis $x = (x_1, \ldots, x_n)$. The following volume formula holds for a simplex $\Delta(P_0, \ldots, P_n)$ with vertices $P_i = (a_{1i}, \ldots, a_{1n})$ with respect to a corresponding affine coordinate system:

$$|\mathrm{vol}(\Delta(P_0, \ldots, P_n))| = \frac{1}{n!} \begin{vmatrix} 1 & 1 & \cdots & 1 \\ a_{10} & a_{11} & \cdots & a_{1n} \\ \vdots & \vdots & & \vdots \\ a_{n0} & a_{n1} & \cdots & a_{nn} \end{vmatrix}.$$

For a proof and further details see [SW89] .

1.3.4 Geometric primitives

The heart of algorithms in computational geometry are geometric primitives, which can be divided into predicates and constructors. Predicates determine the control flow of the algorithm and only need the sign of a polynomial expression, whereas constructors generate new geometric objects and require the value of a polynomial. After a result of [Val79] it is possible to write every algebraic expression of size e as a determinant of order $e + 2$ whose entries are variables or constants. Hence it is possible to formulate geometric primitives as determinants and therefore most decisions in geometric algorithms are based on the signs of determinants.

Let us take a look at some examples for geometric predicates:

Sideness test Given three points $p = (p_x, p_y), q = (q_x, q_y)$, and $r = (r_x, r_y)$ in $\mathbb{R}^2$. The sideness query is: Does the point r lie left, right, or exactly on the line passing through p and q?

How can we formulate this sideness test in terms of a determinant?

What are the conditions that the three point p, q, and r are collinear? Given a line $ax + by + c = 0$, all three points must lie on the same line, which leads to the equations:

$$\begin{aligned} ap_x + bp_y + c = 0 \\ aq_x + bq_y + c = 0 \\ ar_x + bq_y + c = 0. \end{aligned}$$

Writing this system in matrix form, we get

$$\begin{pmatrix} p_x & p_y & 1 \\ q_x & q_y & 1 \\ r_x & r_y & 1 \end{pmatrix} \cdot \begin{pmatrix} a \\ b \\ c \end{pmatrix} = 0.$$

This homogeneous system has a non–trivial solution if and only if

$$D := \begin{vmatrix} p_x & p_y & 1 \\ q_x & q_y & 1 \\ r_x & r_y & 1 \end{vmatrix} = 0.$$

So, p, q and r are collinear iff $D = 0$. The point r is left from the line through p and q iff $D > 0$ and right from the line through p and q iff $D < 0$. Hence, the sideness test can be reduced to the test of the sign of a determinant.

In circle test Given four points $p, q, r, s \in \mathbb{R}^2$, the query is: Does s lie on the circle through p, q and r ? This test is crucial in the computation of Voronoi diagrams for example. We try to transform this query into determinant notation again. A general equation describing a circle K with center $c = (c_x, x_y)$ and radius r is

$$K : (x - c_x)^2 + (y - c_y)^2 = r^2,$$

which can be written as

$$c_x^2 + c_y^2 - r^2 - 2c_x x - 2c_y y + x^2 + y^2 = 0. \tag{1.1}$$

Let us denote $A := c_x^2 + c_y^2 - r^2$, $B := -2c_x$ and $C := -2c_y$. Now we want to find A, B, and C such that all the points p, q, r, s satisfy the equation (1.1) which leads us to the linear system

$$\begin{pmatrix} 1 & p_x & p_y & p_x^2 + p_y^2 \\ 1 & q_x & q_y & q_x^2 + q_y^2 \\ 1 & r_x & r_y & r_x^2 + r_y^2 \\ 1 & s_x & s_y & s_x^2 + r_y^2 \end{pmatrix} \cdot \begin{pmatrix} A \\ B \\ C \\ 1 \end{pmatrix} = 0.$$

Hence, we know that s lies on the circle through p, q, r iff

$$D := \begin{vmatrix} 1 & p_x & p_x & p_x^2 + p_y^2 \\ 1 & q_x & q_y & q_x^2 + q_y^2 \\ 1 & r_x & r_y & r_x^2 + r_y^2 \\ 1 & s_x & s_y & s_x^2 + s_y^2 \end{vmatrix} = 0.$$

A little thought establishes that s lies in the interior of the circle K iff $D > 0$ and in the exterior iff $D < 0$. Of course it is also possible to formulate corresponding determinant tests for arbitrary dimensions.

1.4 General algorithms and practical results

We will briefly present several algorithms for the computation of general integer order determinants and discuss their efficiency distinguishing between rational and polynomial matrix entries.

1.4.1 Standard algorithms

Cofactor expansion We use the expansion theorem of Laplace (see Theorem 2) to receive a first method:

$$|A| = a_{11}|A_{11}| - a_{12}|A_{12}| + \cdots + (-1)^{n+1}a_{1n}|A_{1n}|.$$

Hence, the computation of a determinant of order n involves the computation of n subdeterminants of order $n-1$, as well as n multiplications and n additions, leading to a cost complexity of

$$C(n) = nC(n-1) + n \leq e \cdot n!.$$

Dynamic programming This method essentially employs cofactor expansion, however avoiding successive computation of the same subdeterminants. First we determine the $\binom{n}{2}$ determinants of all 2×2 minors of the first two rows, then the $\binom{n}{3}$ determinants of all 3×3 minors of the first three rows, etc. until the entire determinant is computed.

This leads to a cost complexity of

$$C(n) = \sum_{k=1}^{n} \binom{n}{k} k = n(2^{n-1} - 1).$$

Gaussian Elimination and variants The goal of Gaussian elimination is to transform a matrix A into triangular form using elementary transformations. Gaussian elimination produces the $\Pi \cdot A = L \cdot U$ decomposition (if we also consider possible necessary pivoting) and since L is a lower triangular matrix with maindiagonal 1, U is an upper triangular matrix and Π is a permutation matrix, we have $|A| = \pm|U|$.

Gaussian elimination can be formulated as a triple–loop procedure:

Algorithm

```
sign:=1
for $k = 1$ to $n-1$ do
    Pivot (to avoid division by zero) and update sign
```

```
        for $i = k + 1$ to $n$ do
            for $j = k + 1$ to $n$ do
                $a_{ij} = a_{ij} - (a_{ik}/a_{kk})a_{kj}$
            od
        od
    od
    return sign $\cdot \prod_{l=1}^{n} a_{ll}$.
```

The resulting asymptotic complexity is $O(n^3)$.

One of the disadvantages of Gaussian elimination are the occurring divisions. Cross multiplication eliminates the divisions but leads to an "explosion" of the size of the matrix entries. Bareiss [Bar68] [Bar67] uses the observation of Camille Jordan that the pivoting element of the previous iteration divides each entry to derive fraction free algorithms.

The elimination steps of the different methods are (with $a_{ij}^{(k)}$ denoting the (i, j)–entry in iteration k):

1. Gaussian elimination: $a_{ij}^{(k+1)} = a_{ij}^{(k)} - (a_{ik}^{(k)}/a_{kk}^{(k)})a_{kj}^{(k)}$.
2. Division free elimination: $a_{ij}^{(k+1)} = a_{kk}^{(k)}a_{ij}^{(k)} - a_{ik}^{(k)}a_{kj}^{(k)}$.
3. Fraction free one–step elimination: $a_{ij}^{(k+1)} = (a_{kk}^{(k)}a_{ij}^{(k)} - a_{ik}^{(k)}a_{kj}^{(k)})/a_{k-1,k-1}^{(k-1)}$.
4. Fraction free two–step elimination: see [Bar67]

Modular Arithmetic Here we find an application of a classic algebraic technique. We assume integer entries and choose a set of primes such that the value of the determinant is guaranteed to be smaller than the product of the primes (e.g. using Hadamards inequality to obtain an upper bound for the determinant). Then we evaluate the determinant modulo each of these primes and use the Chinese remainder theorem to reconstruct the original determinant.

The complexity of this approach depends on the used evaluation strategy. Gaussian elimination requires modular inverses which is expensive (although it is plausible to determine modular inverses through table lookup for sufficiently small primes). Dynamic programming avoids modular inverses but requires more operations.

1.4.2 Practical results and the problem of exact computation

Rational entries Comparing the efficiency of the presented determinant algorithms for rational entries it may be observed (see Fortune and Van Wyk [FW93]) that Gaussian elimination, especially the fraction free variants, are superior to minor expansion or dynamic programming for determinants of dense matrices of higher order ($n > 4$), as the asymptotic runtime suggests. Modular techniques can produce similar results as Gaussian elimination.

Talking about practical computation we have to address another topic, the problem of exactness. Recall the use of determinants in geometric algorithms. Geometric predicates determine the control flow and hence have to be evaluated exactly. This means that our determinant computation routine has to produce *exact* results. However, the standard use of floating point arithmetic is not exact (see Goldberg [Gol91]). If the bitlength of the entries (let us consider them to be integers) is large or the matrix dimension is large, then Gaussian elimination (and minor expansion) can fail since the bitlength of intermediate results can exceed the maximal bitlength expressible by the machine. This exactness problem leads to a whole new direction in computational geometry, the field of exact computation. Software packages providing exact computation have been developed with the disadvantage of slow computation speed. Expression compilers and lazy evaluation tried to adapt the precision to the input and obtained good results. There have also been investigations for new algorithms computing the sign of a determinant using floating point operations. Clarkson [Cla92] [BY96] proposed an algorithm first doing a modified Gram–Schmidt orthogonalization and then computing the determinant of a well–conditioned matrix. Following this approach, Avnaim et al. [ABD$^+$97] found a safe method for 2×2 and 3×3 determinants (extended to $n \times n$ determinants by Brönnimann and Yvinec [BY97]), the so–called lattice method.

Polynomial entries The computation of determinants having polynomial entries has to be discussed separately. Since the multiplication (and division) of polynomials cannot be done in constant time, Gaussian elimination or variants are not superior to minor expansion anymore. Especially for sparse matrices, minor expansion leads to faster results. Horowitz and Sahni [HS75] and Sasaki and Murao [SM82] propose special algorithms for polynomial entries and investigate their efficiency.

1.5 Motivation

Now we are familiar with determinants and have seen different algorithms for determinant computation. So why do we want to have determinant formulas for specially structured matrices when we have methods to compute every general determinant ?

Consider the following determinant:

$$V_3 = \begin{vmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{vmatrix}.$$

If we use minor expansion we obtain that

$$V_3 = x_2x_3^2 - x_2^2x_3 - x_1x_3^2 + x_1^2x_3 + x_1x_2^2 - x_1^2x_2. \tag{1.2}$$

Surprisingly, even a computer algebra system like Maple that provides efficient symbolic computation fails to compute the determinant of an equally structured matrix (i.e. $v_{ij} = x_i^{j-1}$) of dimension 8 since the size of the intermediate results cannot be handled. However, the determinant V_n, known as Vandermonde determinant, has the simple formula

$$V_n = \prod_{1 \le i < j \le n} (x_j - x_i). \tag{1.3}$$

If we blindly expand the determinant then we also often loose its structure, like it is not immediately obvious from (1.2) that $V_3 = 0$ for $x_i = x_j$, whereas the representation $V_3 = (x_3 - x_2)(x_2 - x_1)(x_3 - x_1)$ obtained by (1.3) gives immediate insight, offering a clearer simpler structure.

We give another motivating example: Recall the possibility to formulate geometric predicates as determinants. Erickson and Seidel [ES95] discuss the spherical degeneracy problem in $\mathbb{R}^d$. They want to show that the following determinants formulating special in–sphere tests are nonzero:

- Two distinct points $\overline{s_1} = (s_1, 0, \ldots, 0)$ and $\overline{t_1} = (t_1, 0, \ldots, 0)$ on the x_1–axis, one point $\overline{t_i} = t_i \cdot e_i$ from each of the other axes, and one point $\overline{t} = (t, \ldots, t)$ from the main diagonal in $\mathbb{R}^d$:

$$S_1 = \begin{vmatrix} 1 & s_1 & 0 & \cdots & 0 & s_1^2 \\ 1 & t_1 & 0 & \cdots & 0 & t_1^2 \\ 1 & 0 & t_2 & \ddots & \vdots & t_2^2 \\ \vdots & \vdots & \ddots & \ddots & 0 & \vdots \\ 1 & 0 & \cdots & 0 & t_d & t_d^2 \\ 1 & t & \cdots & t & t & dt^2 \end{vmatrix}_{d+2} .$$

- Two distinct points $\overline{t} = (t, \ldots, t)$ and $\overline{s} = (s, \ldots, s)$ from the main diagonal and one point $\overline{t_i} = t_i \cdot e_i$ from each axis in $\mathbb{R}^d$:

$$S_2 = \begin{vmatrix} 1 & t & t & \cdots & t & dt^2 \\ 1 & t_1 & 0 & \cdots & 0 & t_1^2 \\ 1 & 0 & t_2 & \ddots & \vdots & t_2^2 \\ \vdots & \vdots & \ddots & \ddots & 0 & \vdots \\ 1 & 0 & \cdots & 0 & t_d & t_d^2 \\ 1 & s & \cdots & s & s & ds^2 \end{vmatrix}_{d+2} .$$

Showing $S_1 \neq 0$ and $S_2 \neq 0$ for a certain range of the entries (see [ES95]) proves that there are no spherical degeneracies for these examples. Again we are looking for an easy determinant formula such that we can show that the determinant does not vanish for given ranges of the entries.

We see that it is often desirable to know explicit determinant formulas for specially structured matrices of arbitrary order and our aim will be to derive general determinant formulas for a number of important matrix classes such that it is possible to automatically compute a formula for any specified special matrix of these classes.

Chapter 2

Frame forms

In this chapter we will investigate determinants of matrix classes that only have the bordering rows and columns as well as the main diagonal nonzero. We will proceed successively allowing more elements to be nonzero starting with the simplest classes that can be formed. Various interesting examples will illustrate the importance of these matrix classes. We will also show how it is possible to transform matrices with arbitrarily positioned nonzero rows and columns into that form.

The end of the chapter contains a description of the implementation of our results.

2.1 Determinants of 7–matrices

First of all, we will take a look at a matrix class where only the first row, the main diagonal and the first lower side diagonal consist of nonzero elements. The matrices of this class look like the mirrored digit 7. Some interesting properties of determinants of this matrix class can be established.

Definition 3 *A matrix A of order n is called* 7-matrix*, if only the first row, the main diagonal and the first lower side diagonal are nonzero.*

$r : \{1, \dots, n\} \to R$, $r(i) = a_{1i}$ *is called generating function of the first row.*

$d : \{1, \dots, n\} \to R$, $d(i) = a_{ii}$ *is called generating function of the main diagonal.*

$e : \{1, \dots, n-1\} \to R$, $e(i) = a_{i+1,i}$ *is called generating function of the first lower side diagonal.*

Note that we have an overlapping corner element $a_{11} = d(1) = r(1)$ in this setting.

We will refer to the corresponding determinant $|A|$ as 7–determinant *or* 7–form *and often denote it as* $|A| = \textsc{Form7}(n, r, d, e)$.

Example

M is a 7–matrix of order $n+1$ with generating functions $r(i) = a_{n-i+1}$, $(i = 1, \dots, n+1)$, $d(1) = a_n$, $d(i) = x$ $(i = 2, \dots, n)$ and $e(i) = -1$ for $i = 1, \dots, n$.

$$M = \begin{pmatrix} a_n & a_{n-1} & a_{n-2} & \cdots & a_1 & a_0 \\ -1 & x & 0 & \cdots & \cdots & 0 \\ 0 & -1 & x & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & -1 & x & 0 \\ 0 & \cdots & \cdots & 0 & -1 & x \end{pmatrix}. \tag{2.1}$$

We will now prove a general determinant formula for 7–matrices.

Theorem 4 *Let $A \in M(n \times n, R)$ be a 7–matrix with generating functions $r, d,$ and e then*

$$|A| = \sum_{l=1}^{n} (-1)^{l+1} r(l) \prod_{k=1}^{l-1} e(k) \prod_{k=l+1}^{n} d(k). \tag{2.2}$$

Proof.

We will prove identity (2.2) by expansion of the first row:

$$|A| = \sum_{l=1}^{n} (-1)^{l+1} r(l) |A_{1l}|. \tag{2.3}$$

Let $1 \leq l \leq n$, then

$$|A_{1l}| = \begin{vmatrix} e(1) & d(2) & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \ddots & e(l-2) & d(l-1) & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & e(l-1) & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & d(l+1) & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & e(l+1) & d(l+2) & \ddots & \vdots \\ \vdots & & \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 & e(n-1) & d(n) \end{vmatrix}.$$

We see that the main diagonal elements of $|A_{1l}|$ are $e(i)$, $(i = 1, \ldots, l-1)$ and $d(i)$, $(i = l+1, \ldots n)$. If we expand $|A_{1l}|$ and the $l-1$ resulting minors by the first column and the following minors by the first row then we get

$$|A_{1l}| = \prod_{k=1}^{l-1} e(l) \prod_{k=l+1}^{n} d(l). \tag{2.4}$$

Plugging (2.4) in (2.3) implies the theorem.

Note that we could also have proved the theorem expanding the last column and thus getting a simple recurrence but this proof illustrates the resulting formula a little better.

$\square$

Let us apply the theorem to our example (2.1). We get the nice determinant formula

$$|M| = \sum_{l=1}^{n+1} (-1)^{l+1} a_{n-l+1} (-1)^{l-1} x^{n+1-l} \; = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0.$$

Observation

Any polynomial of degree n can be expressed as a 7–determinant of order $n+1$. In fact any binary expression $p(x, y) = a_0 x^n + a_1 x^{n-1} y + \ldots + a_{n-1} x y^{n-1} + a_n y^n$ can be written as 7–determinant of order $n+1$:

$$p(x,y) = \begin{vmatrix} a_0 & a_1 & \cdots & a_{n-1} & a_n \\ y & x & 0 & \cdots & 0 \\ 0 & y & x & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & y & x \end{vmatrix}.$$

Now we will focus on the orientation of 7–matrices. How does the determinant change when we rotate the matrix or when we "mirror" the diagonals. We will see that these minor changes can easily be retransformed into the standard 7–form.

Definition 4 *Let $A^{(l)}$ be a matrix of order n with generating functions $r^{(l)}, d^{(l)}, e^{(l)}$.*

1. *$A^{(1)}$ is called* 1–oriented 7–matrix *if $r^{(1)}(i) = a^{(1)}_{1i}, d^{(1)}(i) = a^{(1)}_{ii}, e^{(1)}(i) = a^{(1)}_{i+1,i}$.*
2. *$A^{(2)}$ is called* 2–oriented 7–matrix *if $r^{(2)}(i) = a^{(2)}_{in}, d^{(2)}(i) = a^{(2)}_{ii}, e^{(2)}(i) = a^{(2)}_{i+1,i}$.*
3. *$A^{(3)}$ is called* 3–oriented 7–matrix *if $r^{(3)}(i) = a^{(3)}_{ni}, d^{(3)}(i) = a^{(3)}_{ii}, e^{(3)}(i) = a^{(3)}_{i,i+1}$.*
4. *$A^{(4)}$ is called* 4–oriented 7–matrix *if $r^{(4)}(i) = a^{(4)}_{i1}, d^{(4)}(i) = a^{(4)}_{ii}, e^{(4)}(i) = a^{(4)}_{i,i+1}$.*
5. *$A^{(5)}$ is called* 5–oriented 7–matrix *if $r^{(5)}(i) = a^{(1)}_{1i}, d^{(5)}(i) = a^{(5)}_{i,n-i+1}, e^{(5)}(i) = a^{(5)}_{i+1,n-i+1}$.*
6. *$A^{(6)}$ is called* 6–oriented 7–matrix *if $r^{(6)}(i) = a^{(6)}_{in}, d^{(6)}(i) = a^{(6)}_{i,n-i+1}, e^{(6)}(i) = a^{(6)}_{i,n-i}$.*
7. *$A^{(7)}$ is called* 7–oriented 7–matrix *if $r^{(7)}(i) = a^{(7)}_{ni}, d^{(7)}(i) = a^{(7)}_{i,n-i+1}, e^{(7)}(i) = a^{(7)}_{i,n-i}$.*
8. *$A^{(8)}$ is called* 8–oriented 7–matrix *if $r^{(8)}(i) = a^{(8)}_{i1}, d^{(8)}(i) = a^{(8)}_{i,n-i+1}, e^{(8)}(i) = a^{(8)}_{i+1,n-i+1}$.*

Note that a 1-oriented 7–matrix is the standard form we discussed above.

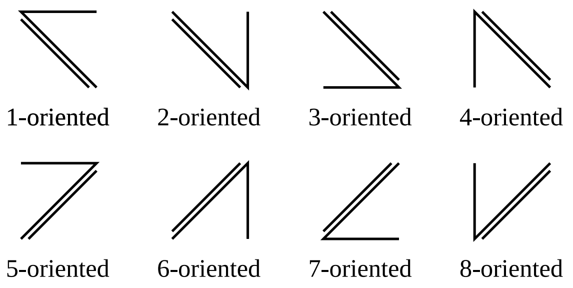

Figure 2: Shapes of oriented 7–forms.

Corollary 1 *Let $A^{(l)}$ be a l–oriented 7–matrix of order n with generating functions $r^{(l)}, d^{(l)}, e^{(l)}$ and define the "reverse" $f^R(i) = f(n-i)$ for a generating function $f : \{1, \ldots, n\} \to R$, then the following identities hold:*

1. $|A^{(1)}| = \textsc{Form7}(n, r^{(1)}, d^{(1)}, e^{(1)})$.
2. $|A^{(2)}| = \textsc{Form7}(n, {r^{(2)}}^R, {d^{(2)}}^R, {e^{(2)}}^R)$.

3. $|A^{(3)}| = \textsc{Form7}(n, {r^{(3)}}^R, {d^{(3)}}^R, {e^{(3)}}^R)$.

4. $|A^{(4)}| = \textsc{Form7}(n, r^{(4)}, d^{(4)}, e^{(4)})$.

5. $|A^{(5)}| = (-1)^{\lfloor\frac{n}{2}\rfloor}\textsc{Form7}(n, {r^{(5)}}^R, d^{(5)}, e^{(5)})$.

6. $|A^{(6)}| = (-1)^{\lfloor\frac{n}{2}\rfloor}\textsc{Form7}(n, r^{(6)}, d^{(6)}, e^{(6)})$.

7. $|A^{(7)}| = (-1)^{\lfloor\frac{n}{2}\rfloor}\textsc{Form7}(n, r^{(7)}, {d^{(7)}}^R, {e^{(7)}}^R)$.

8. $|A^{(8)}| = (-1)^{\lfloor\frac{n}{2}\rfloor}\textsc{Form7}(n, {r^{(8)}}^R, {d^{(8)}}^R, {e^{(8)}}^R)$.

Proof.

1. Follows from definition.
2. We swap column k with column $n-k+1$, for $k = 1, \ldots \lfloor\frac{n}{2}\rfloor$ and get a 8–oriented 7–form. Now we proceed as stated in 8. below.
3. ${}^tA^{(3)} = A^{(2)}$ if $r^{(3)} = r^{(2)}, d^{(3)} = d^{(2)}, e^{(3)} = e^{(2)}$. Alternatively, we could swap the last row $r^{(3)}$ up to the top getting the desired form which would result in a slightly different but equivalent formula.
4. ${}^tA^{(1)} = A^{(4)}$if $r^{(3)} = r^{(2)}, d^{(3)} = d^{(2)}, e^{(3)} = e^{(2)}$.
5. We swap column k with column $n-k+1$, for $k = 1, \ldots \lfloor\frac{n}{2}\rfloor$ and get a 1–oriented 7–form with a reversed row function ${r^{(5)}}^R$.
6. Transposing $A^{(6)}$ yields the 7–oriented 7–form with reversed diagonal functions ${d^{(6)}}^R, {e^{(6)}}^R$. Swapping row k with row $n-k+1$, for $k = 1, \ldots, \lfloor\frac{n}{2}\rfloor$ leaves us with the desired 1–oriented 7–form reversing the diagonal functions another time, thus reaching their original form.
7. As described in 6. we swap row k with row $n-k+1$, for $k = 1, \ldots \lfloor\frac{n}{2}\rfloor$ and get what we want with reversed diagonal functions.
8. Transposing $A^{(8)}$ yields the 5–oriented 7–form with reversed diagonal functions. Using 5. yields the claim.

$\square$

In the following we will denote an s–oriented 7–form of order n with generating functions r, d, e with the function call

$$\textsc{Form7}_s(n, r, d, e).$$

2.2 Determinants of arrow matrices

In this section we are investigating the determinant of arrow shaped matrices. We will also show how we can compute a variant of arrow determinants using determinants of 7–matrices.

Definition 5 *An $n \times n$ matrix A is called an* arrow matrix, *if only the first row, the first column and the main diagonal consist of nonzero entries.*

$r : \{1, \ldots, n\} \to R$, $r(i) = a_{1i}$ *is called generating function of the first row.*

$c : \{1, \ldots, n\} \to R$, $c(i) = a_{i1}$ *is called generating function of the first column.*

$d : \{1 \dots , n\} \to R$, $d(i) = a_{ii}$ is called generating function of the main diagonal.

Note that we have an overlapping corner element $a_{11} = d(1) = r(1) = c(1)$ in this setting.

The corresponding determinant $|A|$ will be called arrow determinant *or* arrow form *and will be often denoted as* $\textsc{Arrow}(n, r, c, d)$.

Example

$$A = \begin{pmatrix} d_1 & r_2 & r_3 & \cdots & r_n \\ c_2 & d_2 & 0 & \cdots & 0 \\ c_3 & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & d_{n-1} & 0 \\ c_n & 0 & \cdots & 0 & d_n \end{pmatrix}$$

A is an arrow matrix of order n with diagonal generating function $d(i) = d_i$, row generating function $r(1) = d_1$, $r(i) = r_i$ (for $i = 2, \dots , n$) and column generating function $c(1) = d_1$, $c(i) = c_i$ (for $i = 2, \dots , n$).

Now we will derive a general determinant formula for arrow matrices. We will try to use minor expansion with the goal to reach trivial minors (like diagonal minors) as soon as possible.

Theorem 5 *The determinant of an arrow matrix A of order n with generating functions r, c and d is*

$$|A| = \prod_{l=1}^{n} d(l) - \sum_{l=2}^{n} r(l)c(l) \prod_{\substack{k=2 \\ k \neq l}}^{n} d(k). \tag{2.5}$$

Proof.

We will prove identity (2.5) by expansion of the first row of A:

$$|A| = \sum_{l=1}^{n} (-1)^{l+1} a_{1l} |A_{1l}| = \sum_{l=1}^{n} (-1)^{l+1} r(l) |A_{1l}|. \tag{2.6}$$

Assume that we have $2 \le l \le n$, then

$$|A_{1l}| = \begin{vmatrix} c(2) & d(2) & 0 & \cdots & 0 & 0 & \cdots & 0 \\ c(3) & 0 & d(3) & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 & 0 & \cdots & 0 \\ c(l-1) & 0 & \cdots & 0 & d(l-1) & 0 & \cdots & \\ c(l) & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ c(l+1) & 0 & \cdots & 0 & 0 & d(l+1) & \ddots & \vdots \\ \vdots & \vdots & & \vdots & \vdots & \ddots & \ddots & 0 \\ c(n) & 0 & \cdots & 0 & 0 & \cdots & 0 & d(n) \end{vmatrix}.$$

Swapping row l up to the top, we get

$$|A_{1l}| = (-1)^{l-2} \begin{vmatrix} c(l) & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ c(2) & d(2) & \ddots & & & & \vdots \\ \vdots & 0 & \ddots & \ddots & & & \vdots \\ c(l-1) & \vdots & \ddots & d(l-1) & \ddots & & \vdots \\ c(l+1) & \vdots & & \ddots & d(l+1) & \ddots & \vdots \\ \vdots & \vdots & & & \ddots & \ddots & 0 \\ c(n) & 0 & \cdots & \cdots & \cdots & 0 & d(n) \end{vmatrix}$$

$$= (-1)^{l-2} c(l) \prod_{\substack{k=2 \\ k \neq l}}^{n} d(k). \tag{2.7}$$

Substituting (2.7) back in (2.6) we get

$$|A| = d(1)|A_{11}| + \sum_{l=2}^{n} (-1)^{l+1} r(l)(-1)^{l-2} c(l) \prod_{\substack{k=2 \\ k \neq l}}^{n} d(k) = d(1)|A_{11}| - \sum_{l=2}^{n} r(l)c(l) \prod_{\substack{k=2 \\ k \neq l}}^{n} d(k).$$

Since $|A_{11}|$ is a diagonal minor, we have $|A_{11}| = \prod_{l=2}^{n} d(l)$ and the theorem follows.

Note that we could also have proved this by solving a simple recurrence formula but our proof here is a little more illustrative.

$\square$

At the moment we are only dealing with arrows pointing to the upper left corner of the matrix (i.e. the arrow head is a_{11}). How does the determinant of an arrow shaped matrix change when the arrow points at other corners of the matrix?

To deal with this issue we define the orientation of an arrow matrix.

Definition 6 *Enumerate the corner elements of the matrix clockwise starting with a_{11}, i.e.*

$$corner_1 := a_{11},\ corner_2 := a_{1n},\ corner_3 := a_{nn},\ corner_4 := a_{n1}.$$

*An arrow shaped matrix A is called s–*oriented arrow matrix *if the arrow head is $corner_s$.*

The diagonal-, row- and column generating functions $r^{(s)}, c^{(s)}, d^{(s)}$ of an s–oriented arrow matrix A are again defined such that their values coincide in the arrow head.

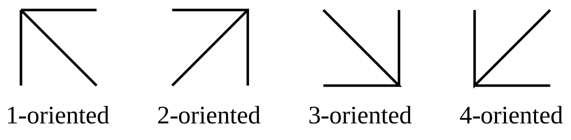

Figure 3: Shapes of oriented arrow forms.

Example

$|A|$ is a 4-oriented arrow determinant:

$$|A| = \left| \begin{array}{cccccc} c(1) & 0 & \cdots & \cdots & 0 & d(1) \\ c(2) & 0 & & \cdot & d(2) & 0 \\ \vdots & \vdots & \cdot & \cdot & \cdot & \vdots \\ c(n-2) & 0 & d(n-2) & \cdot & & \vdots \\ c(n-1) & d(n-1) & 0 & \cdots & 0 & 0 \\ d(n) & r(2) & r(3) & \cdots & r(n-1) & r(n) \end{array} \right|$$

Note that 1–oriented arrow determinants are the standard form which we already discussed above.

Now we try to obtain similar results for the determinant of s–oriented arrow matrices $(s = 1, 2, 3, 4)$.

Corollary 2 *Let $A^{(s)}$ be an s–oriented arrow matrix of order n with generating functions $r^{(s)}, c^{(s)}, d^{(s)}$ for row, column and diagonal respectively, then the following identities hold:*

1. $|A^{(1)}| = \textsc{Arrow}(n, r^{(1)}, c^{(1)}, d^{(1)})$.
2. $|A^{(2)}| = (-1)^{\lfloor \frac{n}{2} \rfloor} \textsc{Arrow}(n, {r^{(2)}}^R, c^{(2)}, d^{(2)})$.
3. $|A^{(3)}| = \textsc{Arrow}(n, {r^{(3)}}^R, {c^{(3)}}^R, {d^{(3)}}^R)$.
4. $|A^{(4)}| = (-1)^{\lfloor \frac{n}{2} \rfloor} \textsc{Arrow}(n, r^{(4)}, {c^{(4)}}^R, {d^{(4)}}^R)$.

Proof.

1. This follows from identity (2.5).
2. If we swap column k with column $n - k + 1$, for $k = 1, \ldots \lfloor \frac{n}{2} \rfloor$, we get the desired 1–oriented arrow form with the reversed row function ${r^{(2)}}^R$.
3. In this case we proceed in two steps: First we swap column k with column $n-k+1$, for $k = 1, \ldots \lfloor \frac{n}{2} \rfloor$, receiving a 4–oriented arrow form with reversed row function, then we swap row k with row $n-k+1$, for $k = 1, \ldots \lfloor \frac{n}{2} \rfloor$ and obtain the 1–oriented arrow form with reversed row and column functions ${r^{(3)}}^R, {c^{(3)}}^R$ and reversed main diagonal function ${d^{(3)}}^R$. The sign factor cancels out.
4. Here, we simply swap row k with row $n - k + 1$, for $k = 1, \ldots \lfloor \frac{n}{2} \rfloor$ and get the desired form with reversed column and diagonal functions ${c^{(4)}}^R, {d^{(4)}}^R$.

$\square$

In the future we will denote an s–oriented arrow form of order n with generating functions r, c, d with the function call

$$\textsc{Arrow}_s(n, r, c, d).$$

Presently, the shaft of an arrow shaped matrix was always the main diagonal or counter main diagonal. Moving the arrow shaft diagonally, i.e. from the main diagonal to one of the side diagonals, simplifies the determinant. If the shaft is moved diagonally for more than one position, the determinant becomes zero. Moving one position, obvious clever expansion results in the product of the diagonal elements multiplied by the last element of the row or column with a possible sign factor.

If we also allow the first upper or lower side diagonal to be nonzero, we are speaking of *fat arrow matrices.* It will be shown how their determinant formula can be computed with the help of 7–determinants.

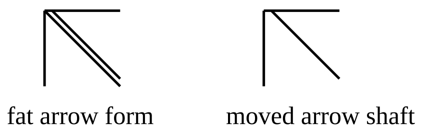

Figure 4: Special arrow forms.

Theorem 6 *Let A be a fat arrow matrix of order n with generating functions r, c, d and function $e : \{1, \dots, n-1\}$, $e(i) = a_{i,i+1}$ generating the first upper side diagonal, then*

$$|A| = \prod_{l=1}^{n} d(l) + \sum_{l=2}^{n} (-1)^{l+1} c(l) \prod_{k=l+1}^{n} d(k) \text{ Form7}(l-1, r_{|2..l}, e, d_{|2..l-1}), \tag{2.8}$$

where $f_{|p_1..p_2}$ simply means the function f restricted to the range $\{p_1, \dots, p_2\}$.
The determinant of a fat arrow matrix will be denoted as $|A| = \text{FatArrow}(n, r, c, d, e)$.

Proof.

We expand the first column:

$$|A| = \sum_{l=1}^{n} (-1)^{(l+1)} c(l) |A_{l1}|.$$

Consider $2 \leq l \leq n$, then we have:

$$|A_{l1}| = \begin{vmatrix} r(2) & \cdots & r(l-1) & r(l) & r(l+1) & \cdots & r(n-1) & r(n) \\ d(2) & e(2) & 0 & \cdots & 0 & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots & & & \vdots \\ \vdots & \ddots & d(l-1) & e(l-1) & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & d(l+1) & e(l+1) & \ddots & \vdots \\ \vdots & & \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \vdots & \vdots & & \ddots & d(n-1) & e(n-1) \\ 0 & \cdots & 0 & 0 & \cdots & \cdots & 0 & d(n) \end{vmatrix}.$$

We see that the submatrix generated by rows $l+1, \dots, n$ and columns $l+1, \dots, n$ is an upper triangular matrix, hence

$$|A_{1l}| = \prod_{k=l+1}^{n} d(k) \begin{vmatrix} r(2) & r(2) & r(3) & \cdots & r(l) \\ d(2) & e(2) & 0 & \cdots & 0 \\ 0 & d(3) & e(3) & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & d(l-1) & e(l-1) \end{vmatrix}_{l-1}.$$

The determinant above is the desired 7–determinant of order $l-1$. The special construction of this 7–determinant makes it obvious that substituting l for n in the corresponding determinant of order $n-1$ yields what we want.

The function r from $2,\dots n$ is the row generating function of the 7–determinant, the function e plays the role of the main diagonal generating function and d from $2,\dots n-1$ is the generating function of the first lower side diagonal. The theorem follows.

$\square$

We observe that it is sufficient to compute the determinant of the 7–matrix of order $n-1$ only once.

It remains to note that it is possible to transform fat arrow matrices of different orientation or side diagonal position into the previous form using the same methods as in Corollary 2.

2.3 Determinants of N–matrices

In this section we will focus on another class of matrices that only have the main diagonal and two bordering rows or columns nonzero. Note that arrow matrices had one bordering row and one bordering column nonzero. We will now examine the effect that this minor shape change has on the determinant.

Definition 7 *A matrix A of order n is called* N–matrix, *if only elements of the first column, the last column and the main diagonal are nonzero.*

$c_1 : \{1,\dots,n\} \to R$, $c_1(i) = a_{i1}$ *is called generating function of column* 1.

$c_n : \{1,\dots,n\} \to R$, $c_n(i) = a_{in}$ *is called generating function of column n.*

$d : \{1,\dots,n\} \to R$, $d(i) = a_{ii}$ *is called main diagonal generating function.*

Note that we have overlapping corner elements $a_{11} = c_1(1) = d(1)$ and $a_{nn} = c_n(n) = d(n)$ in this setting.

The determinant of an N–matrix will be referred to as N–determinant *or* N–form *and will be often denoted as* $\textsc{Nform}(n, c_1, c_2, d)$.

Example

A is an N–matrix of order n with diagonal generating function $d(1) = a_1, d(n) = b_n, d(i) = d_i$ (for $i = 2,\dots,n-1$) and generating functions $c_1(i) = a_i$ and $c_n(i) = b_i$ for column 1 and n respectively:

$$\begin{pmatrix} a_1 & 0 & 0 & \cdots & 0 & b_1 \\ a_2 & d_2 & 0 & & 0 & b_2 \\ a_3 & 0 & d_3 & \ddots & \vdots & b_3 \\ \vdots & \vdots & \ddots & \ddots & 0 & \vdots \\ a_{n-1} & 0 & & 0 & d_{n-1} & b_{n-1} \\ a_n & 0 & \cdots & 0 & 0 & b_n \end{pmatrix}.$$

We will see now that the determinant of this matrix class is even simpler than the determinant of arrow matrices.

Theorem 7 *The determinant of a N–matrix A of order n with column generating functions c_1 and c_n and main diagonal generating function d is*

$$|A| = (c_1(1)c_n(n) - c_n(1)c_1(n)) \prod_{l=2}^{n-1} d(l).$$

Proof.

We expand A by the first row:

$$|A| = c_1(1)|A_{11}| + (-1)^{n+1}c_n(1)|A_{1n}|.$$

It follows immediately that $|A_{11}| = c_n(n)\prod_{l=2}^{n-1} d(l)$. Examining $|A_{1n}|$, it is obvious that after swapping the last row up to the top ($n-2$ row exchanges) we are left with a lower triangular matrix. Hence $|A_{1n}| = (-1)^{n-2}c_1(n)\prod_{l=2}^{n-1} d(l)$ and the theorem follows.

$\square$

After having established this identity for N–forms we want to generalize it for slight modifications of this matrix class (like in the case of oriented arrow matrices).

Definition 8 *Let $A^{(l)}$ be a matrix of order n with generating functions $f^{(l)}, g^{(l)}$, and $d^{(l)}$.*

1. *$A^{(1)}$ is called* 1–oriented N–matrix *if $f^{(1)}(i) = a_{i1}, g^{(1)}(i) = a_{in}, d^{(1)}(i) = a_{ii}$.*
2. *$A^{(2)}$ is called* 2–oriented N–matrix *if $f^{(2)}(i) = a_{1i}, g^{(2)}(i) = a_{ni}, d^{(2)}(i) = a_{ii}$.*
3. *$A^{(3)}$ is called* 3–oriented N–matrix *if $f^{(3)}(i) = a_{i1}, g^{(3)}(i) = a_{in}, d^{(3)}(i) = a_{i,n-i+1}$.*
4. *$A^{(4)}$ is called* 4–oriented N–matrix *if $f^{(4)}(i) = a_{1i}, g^{(4)}(i) = a_{ni}, d^{(4)}(i) = a_{i,n-i+1}$.*

Note that the generating functions are overlapping again and that the 1–oriented N–matrix is the standard N–matrix discussed above.

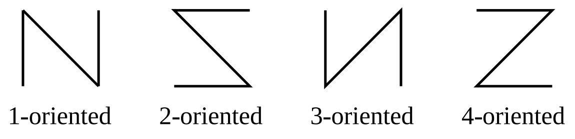

Figure 5: Shapes of oriented N–forms.

Example

Determinant of a 4-oriented N–matrix A of order n with $f^{(4)}(i) = f_i, g^{(4)}(i) = g_i, d^{(4)}(i) = d_i$:

$$|A| = \begin{vmatrix} f_1 & f_2 & f_3 & \cdots & f_{n-1} & f_n \\ 0 & \cdots & \cdots & 0 & d_2 & 0 \\ \vdots & & \cdot & \cdot & \cdot & \vdots \\ \vdots & \cdot & d_{n-2} & \cdot & & \vdots \\ 0 & d_{n-1} & 0 & \cdots & \cdots & 0 \\ g_1 & g_2 & g_3 & \cdots & g_{n-1} & g_n \end{vmatrix}.$$

We will now show how determinant formulas for all orientations of N–matrices can be obtained.

Corollary 3 *Let $A^{(l)}$ be an l–oriented N–matrix of order n with generating functions $f^{(l)}, g^{(l)}, d^{(l)}$, then the following identities hold:*

1. $|A^{(1)}| = \textsc{Nform}(n, f^{(1)}, g^{(1)}, d^{(1)}).)$

2. $|A^{(2)}| = \textsc{Nform}(n, f^{(2)}, g^{(2)}, d^{(2)}).$

3. $|A^{(3)}| = (-1)^{\lfloor \frac{n}{2} \rfloor}\textsc{Nform}(n, g^{(3)}, f^{(3)}, d^{(3)}).$

4. $|A^{(4)} = (-1)^{\lfloor \frac{n}{2} \rfloor}\textsc{Nform}(n, g^{(4)}, f^{(4)}, {d^{(4)}}^R).$

Proof.

1. This follows from Theorem 7 with $d = d^{(1)}, c_1 = f^{(1)}, c_n = g^{(1)}$.
2. Since $|{}^tA| = |A|$, this follows again from Theorem 7 with $d = d^{(2)}, c_1 = f^{(2)}, c_n = g^{(2)}$.
3. If we swap column k with column $n-k+1$ for $k = 1, \dots, \lfloor \frac{n}{2} \rfloor$ then we obtain a 1–oriented N–form and the claim follows from 1. with $f^{(1)} = g^{(3)}$ and $g^{(1)} = f^{(3)}$.
4. Follows from 3. with $f^{(3)} = f^{(4)}, g^{(3)} = g^{(4)}$, and $d^{(3)} = {d^{(4)}}^R$ since $|{}^tA| = |A|$.

$\square$

Having the first upper or lower side diagonal nonzero instead of the main diagonal, it is straightforward to see that swapping the last row up to the top (or the first row down to the bottom, respectively) we obtain the normal N–form. If the nonzero diagonal is beyond that, the determinant will be zero.

How does the determinant of a N shaped matrix change, if we introduce another neighbouring nonzero diagonal? It will be shown that an easy reduction to determinants of 7–matrices can be made. For that purpose we examine modified 2–oriented N–forms.

Corollary 4 *Let A be a 2–oriented N–matrix of order n with generating functions f, g, d and additional function $e(i) = a_{i+1,i}$ $i = 1, \dots, n-1$ generating the first lower side diagonal, then*

$$|A| = g(n)\textsc{Form7}(n-1, f_{|1..n-1}, d_{|1..n-1}, e_{|1..n-2}) - f(n)\textsc{Form7}(n-1, g_{|1..n-1}, d_{|^{1=g(1)}_{1..n-1}}, e_{|1..n-2}),$$

with

$$f_{|^{1=g(k)}_{1..n-1}}(i) = \begin{cases} g(k) & \mathit{if}\, i = 1 \\ f(i) & \mathit{if}\, 2 \le i \le n-1. \end{cases}$$

Proof.

Expanding the last column of A we get $|A| = g(n)|A_{nn}| + (-1)^{n+1}f(n)|A_{1n}|$.

$|A_{nn}|$ obviously is the first claimed 7–form and swapping row $n-1$ of $|A_{1n}|$ up to the top we get the desired second 7–form while the sign factor reduces to -1.

$\square$

We will call matrices of this type *fat N–matrices.* Observe that it is straightforward to transform fat N–matrices of other orientations into this standard form.

2.4 Determinants of R–matrices

In the two previous sections we discussed determinants of matrix classes that only had two bordering rows or columns and the main diagonal nonzero. Now we are interested in allowing another bordering row or column to be nonzero.

Definition 9 *A matrix A of order n is called* R–matrix*, if only the first and last column, the first row and the main diagonal are nonzero.*

$c_1 : \{1, \dots, n\} \to K, f(i) = a_{i1}$ *is called generating function of column 1.*

$c_2 : \{1 \dots, n\} \to K, g(i) = a_{in}$ *is called generating function of column n.*

$r : \{1, \dots, n\} \to K, h(i) = a_{1i}$ *is called generating function of row* 1*.*

$d : \{1, \dots, n\} \to K, d(i) = a_{ii}$ *is called generating function of the main diagonal.*

Note that we have overlapping corner elements $a_{11} = r(1) = c_1(1) = d(1), a_{1n} = r(n) = c_2(1)$ and $a_{nn} = d(n) = c_2(n)$ in this setting.

The determinant of an R–matrix will be referred to as R–determinant *or* R–form *and will be often denoted as* $\textsc{Rform}(n, c_1, c_2, r, d)$.

Example

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 5 \\ 3 & 2 & 0 & 0 & 5 \\ 5 & 0 & 3 & 0 & 5 \\ 7 & 0 & 0 & 4 & 5 \\ 9 & 0 & 0 & 0 & 5 \end{pmatrix}.$$

Now we want to derive a general determinant formula for R–matrices.

Theorem 8 *Let A be an R–matrix of order n with generating functions c_1, c_2, r, d, then we have*

$$|A| = -c_1(n)\textsc{Arrow}(n-1, r_{|^{1=c_2(1)}_{1..n-1}}, c_{2|_{1..n-1}}, d_{|^{1=c_2(1)}_{1..n-1}}) + c_2(n)\textsc{Arrow}(n-1, r_{|_{1..n-1}}, c_{1|_{1..n-1}}, d_{|_{1..n-1}}) \quad (2.9)$$

where $f_{|_{1..n-1}}$simply means the function f limited to the range $\{1, \dots, n-1\}$ and

$$f_{|^{1=g(k)}_{1..n-1}}(i) = \begin{cases} g(k) & \mathit{if}\, i = 1 \\ f(i) & \mathit{if}\, 2 \le i \le n-1. \end{cases}$$

Proof.

We will prove identity (2.9) by expansion of the last row of A:

$$|A| = (-1)^{n+1} c_1(n)|A_{n1}| + c_2(n)|A_{nn}|,$$

where

$$|A_{n1}| = \begin{vmatrix} r(2) & r(3) & \cdots & r(n-1) & c_2(1) \\ d(2) & 0 & \cdots & 0 & c_2(2) \\ 0 & d(3) & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & d(n-1) & c_2(n-1) \end{vmatrix} \tag{2.10}$$

and

$$|A_{nn}| = \begin{vmatrix} c_1(1) & r(2) & r(3) & \cdots & r(n-1) \\ c_1(2) & d(2) & 0 & \cdots & 0 \\ c_1(3) & 0 & d(3) & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ c_1(n-1) & 0 & \cdots & 0 & d(n-1) \end{vmatrix}. \tag{2.11}$$

It is obvious from (2.11) that $|A_{nn}| = \text{Arrow}(n-1, r_{|_{1..n-1}}, c_{1|_{1..n-1}}, d_{|_{1..n-1}})$.
After swapping the last column $n-1$ through to column 1 in (2.10), we see that

$$|A_{n1}| = (-1)^{n-2}\text{Arrow}(n-1, r_{|^{1=c_2(1)}_{1..n-1}}, c_{2|_{1..n-1}}, d_{|^{1=c_2(1)}_{1..n-1}}),$$

which implies the theorem.

$\square$

There are several other ways to pick three nonzero bordering rows or columns and a main diagonal to get matrices that are slight modifications of R–matrices. In the following we define oriented R–matrices to take this matter into account and show how their determinants differ from those of normal R–matrices.

Definition 10 *Let $A^{(l)}$ be a matrix of order n with generating functions $f,^{(l)}$ $g^{(l)}, h^{(l)}$ and $d^{(l)}$.*

1. *$A^{(1)}$ is called* 1–oriented R–matrix *if* $f^{(1)}(i) = a^{(1)}_{i1}, g^{(1)}(i) = a^{(1)}_{in}, h^{(1)}(i) = a^{(1)}_{1i}, d^{(1)}(i) = a^{(1)}_{ii}$.
2. *$A^{(2)}$ is called* 2–oriented R–matrix *if* $f^{(2)}(i) = a^{(2)}_{1i}, g^{(2)}(i) = a^{(2)}_{ni}, h^{(2)}(i) = a^{(2)}_{in}, d^{(2)}(i) = a^{(2)}_{ii}$.
3. *$A^{(3)}$ is called* 3–oriented R–matrix *if* $f^{(3)}(i) = a^{(3)}_{i1}, g^{(3)}(i) = a^{(3)}_{in}, h^{(3)}(i) = a^{(3)}_{ni}, d^{(3)}(i) = a^{(3)}_{ii}$.
4. *$A^{(4)}$ is called* 4–oriented R–matrix *if* $f^{(4)}(i) = a^{(4)}_{1i}, g^{(4)}(i) = a^{(4)}_{ni}, h^{(4)}(i) = a^{(4)}_{i1}, d^{(4)}(i) = a^{(4)}_{ii}$.
5. *$A^{(6)}$ is called* 5–oriented R–matrix *if* $f^{(5)}(i) = a^{(5)}_{i1}, g^{(5)}(i) = a^{(5)}_{in}, h^{(5)}(i) = a^{(5)}_{1i}, d^{(5)}(i) = a^{(5)}_{i,n-i+1}$.
6. *$A^{(5)}$ is called* 6–oriented R–matrix *if* $f^{(6)}(i) = a^{(6)}_{1i}, g^{(6)}(i) = a^{(6)}_{ni}, h^{(6)}(i) = a^{(6)}_{in}, d^{(6)}(i) = a^{(6)}_{i,n-i+1}$.
7. *$A^{(7)}$ is called* 7–oriented R–matrix *if* $f^{(7)}(i) = a^{(7)}_{i1}, g^{(7)}(i) = a^{(7)}_{in}, h^{(7)}(i) = a^{(7)}_{ni}, d^{(7)}(i) = a^{(7)}_{i,n-i+1}$.
8. *$A^{(8)}$ is called* 8–oriented R–matrix *if* $f^{(8)}(i) = a^{(8)}_{1i}, g^{(8)}(i) = a^{(8)}_{ni}, h^{(8)}(i) = a^{(8)}_{i1}, d^{(8)}(i) = a^{(8)}_{i,n-i+1}$.

Note that the generating functions are overlapping again and that the 1–oriented R–matrix is the standard R–matrix discussed above.

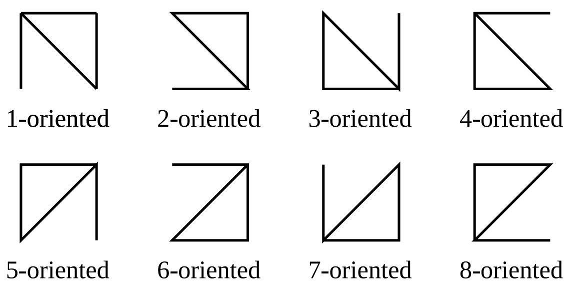

Figure 6: Shapes of oriented R–forms.

Corollary 5 *Let $A^{(l)}$ be an l–oriented R–matrix of order n with generating functions $f^{(l)}, g^{(l)}, h^{(l)}, d^{(l)}$, then the following identities hold:*

1. $|A^{(1)}| = \text{RFORM}(n, f^{(1)}, g^{(1)}, h^{(1)}, d^{(1)}).$

2. $|A^{(2)}| = \text{RFORM}(n, {g^{(2)}}^R, {f^{(2)}}^R, {h^{(2)}}^R, {d^{(2)}}^R).$

3. $|A^{(3)}| = \text{RFORM}(n, {g^{(3)}}^R, {f^{(3)}}^R, {h^{(3)}}^R, {d^{(3)}}^R).$

4. $|A^{(4)}| = \text{RFORM}(n, f^{(4)}, g^{(4)}, h^{(4)}, d^{(4)}).$

5. $|A^{(5)}| = (-1)^{\lfloor \frac{n}{2} \rfloor} \text{RFORM}(n, g^{(5)}, f^{(5)}, {h^{(5)}}^R, d^{(5)}).$

6. $|A^{(6)}| = (-1)^{\lfloor \frac{n}{2} \rfloor} \text{RFORM}(n, {f^{(6)}}^R, {g^{(6)}}^R, h^{(6)}, d^{(6)}).$

7. $|A^{(7)}| = (-1)^{\lfloor \frac{n}{2} \rfloor} \text{RFORM}(n, {f^{(7)}}^R, {g^{(7)}}^R, h^{(7)}, {d^{(7)}}^R).$

8. $|A^{(8)}| = (-1)^{\lfloor \frac{n}{2} \rfloor} \text{RFORM}(n, g^{(8)}, f^{(8)}, {h^{(8)}}^R, {d^{(8)}}^R).$

Proof.

1. Follows from Theorem 8.

2. Follows from 3. since ${}^tA^{(2)} = A^{(3)}$ for identical generating functions.

3. Swapping row k with row $n-k+1$, for $k = 1, \dots \lfloor \frac{n}{2} \rfloor$, we obtain a 5–oriented R–form with reversed generating functions $f^{(3)}, g^{(3)}, d^{(3)}$. Now we swap column k with column $n-k+1$, for $k = 1, \dots \lfloor \frac{n}{2} \rfloor$ and get the standard form with also $h^{(4)}$ reversed and f and g exchanged.

4. Follows from ${}^tA^{(4)} = A^{(1)}$.

5. Swapping column k with column $n-k+1$, for $k = 1, \dots \lfloor \frac{n}{2} \rfloor$ does the trick, getting a reversed $h^{(5)}$ and exchanged $f^{(5)}$ and $g^{(5)}$.

6. Transposing leads us to a 7–oriented R–form with transposed $d^{(6)}$.

7. Swap row k with row $n-k+1$, for $k = 1, \dots \lfloor \frac{n}{2} \rfloor$ and get the desired form with reversed $f^{(7)}, g^{(7)}$.

8. Transpose and get 5–oriented R–form with reversed $d^{(8)}$.

$\square$

In the future we will denote an s–oriented R–form of order n with generating functions f, g, h, d with the function call

$$\textsc{Rform}_s(n, f, g, h, d).$$

After investigating the standard R–form in all orientations we will have a look at minor changes. Consider moving the diagonal again: As before, we are left with a zero determinant if we move more than one position diagonally. Moving only one position diagonally, clever expansion results in a triangular matrix and an arrow form which leads to an easy determinant formula.

Now we will study how the introduction of a nonzero side diagonal changes the determinant of such matrices.

Unfortunately it is necessary to distinguish the case of introducing a nonzero first upper side diagonal, called *fat–1 R–matrix*, and the case of introducing a nonzero first lower side diagonal, called *fat–2 R–matrix*, as we will see in the following.

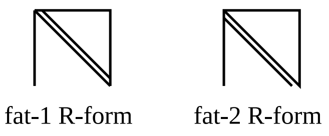

Figure 7: Shapes of fat R-forms.

Corollary 6 *Let A be an R–matrix of order n with generating functions c_1, c_2, r, d and additional generating function $e(i) = a_{i,i+1}$, called fat–1 R–matrix, then we have*

$$\begin{aligned} |A| \;=\;& g(n)\textsc{FatArrow}(n-1, r_{|_{1..n-1}}, c_{1|_{1..n-1}}, d_{|_{1..n-1}}, e_{|_{1..n-2}}) \\ & -f(n)\textsc{FatArrow}(n-1, r_{|^{1=c_2(1)}_{1..n-1}}, c_{2|_{1..n-1}}, d_{|^{1=c_2(1)}_{1..n-1}}, e_{|_{1..n-2}}). \end{aligned}$$

Proof.

Expanding the last row of A yields $|A| = (-1)^{n+1} c_1(n)|A_{n1}| + c_2(n)|A_{nn}|$.

$|A_{nn}|$ is already the needed fat arrow form and swapping the last column of $|A_{n1}|$ through to column 1 results in the second fat arrow form and a simplification of the sign factor to -1.

$\square$

This was an easy reduction to fat arrow matrices. However, the next case will produce a rather involved formula.

Corollary 7 *Let A be an R–matrix of order n with generating functions c_1, c_2, r, d and additional generating function $e : \{1, \ldots, n-1\} \to R$, $e(i) = a_{i+1,i}$, called fat–2 R–matrix, then we have*

$$\begin{aligned} |A| \;=:\;& \textsc{Fat2Rform}(n, c_1, c_2, r, d) \\ =\;& c_2(n)\textsc{FatArrow}(n-1, c_{1|_{1..n-1}}, r_{|_{1..n-1}}, d_{|_{1..n-1}}, e_{|_{1..n-2}}) \\ & -c_1(n)\textsc{FatArrow}(n-1, c_{2|_{1..n-1}}, r_{|^{1=C_2(1)}_{1..n-1}}), d_{|^{1=c_2(1)}_{1..n-1}}, e_{|^{1=c_2(2)}_{1..n-2}}) \\ & -e(n-1)\textsc{Fat2Rform}(n-1, c_{1|_{1..n-1}}, c_{2|_{1..n-1}}, r_{|^{n-1=c_2(1)}_{1..n-1}}, d_{|^{n-1=c_2(n-1)}_{1..n-1}}, e_{|_{1..n-2}}). \end{aligned}$$

Proof.

We expand the last row of A :

$$|A| = (-1)^{n+1}c_1(n)|A_{n1}| + (-1)^{2n-1}e(n-1)|A_{n,n-1}| + c_2(n)|A_{nn}|.$$

Observe that $|A_{nn}| = \textsc{FatArrow}(n-1, c_{1|_{1..n-1}}, r_{|_{1..n-1}}, d_{|_{1..n-1}}, e_{|_{1..n-2}})$ as above if we transpose the minor. Moreover, swapping the last column through to the first column and transposing the result, we establish that $|A_{n1}| = (-1)^{n-2}\textsc{FatArrow}(n-1, c_{2|_{1..n-1}}, r_{|_{1..n-1}^{1=c_2(1)}}, d_{|_{1..n-1}^{1=c_2(1)}}, e_{|_{1..n-2}^{1=c_2(2)}})$. The rest is straightforward to see.

$\square$

The last determinant formula can be viewed as a recurrence. Unfortunately we cannot solve this recurrence in general, so we still lack an explicit formula for this case.

It remains to note that it is possible to reduce fat R–matrices of other orientations to one of the previous cases using similar methods as in Corollary 5.

2.5 Determinants of DB–matrices

It is obvious that we now want to make the final step, that is allowing all bordering elements and the main diagonal elements to be nonzero. The resulting matrix class looks like a box crossed with one diagonal line which is exactely the logo of the "Deutsche Bank" (hence the name DB–matrix).

Definition 11 *A matrix A of order n is called* DB–matrix *if only the first and last rows and columns as well as the main diagonal are nonzero.*

$c_1 : \{1, \dots, n\} \to R,\ c_1(i) = a_{i1}$ *is called generating function of column 1.*

$c_2 : \{1, \dots, n\} \to R,\ c_2(i) = a_{in}$ *is called generating function of column n.*

$r_1 : \{1, \dots, n\} \to R,\ r_1(i) = a_{1i}$ *is called generating function of row* 1.

$r_2 : \{1, \dots, n\} \to R,\ r_2(i) = a_{ni}$ *is called generating function of row n.*

$d : \{1, \dots, n\} \to R,\ d(i) = a_{ii}$ *is called generating function of the main diagonal.*

The matrix corner elements $a_{11}, a_{1n}, a_{n1}, a_{nn}$ will be denoted by $c_1(1), c_2(1), c_1(n), c_2(n)$ respectively, as they are overlapping again.

The determinant of a DB–matrix will be referred to as DB–determinant *or* DB–form *and will be often denoted as* $\textsc{DBform}(n, c_1, c_2, r_1, r_2, d)$.

Example

Recall the examples from the first chapter concerned with the detection of spherical degeneracies[ES95]. The mentioned in–sphere predicates S_1 and S_2 are in DB–form.

$$S_1 = \begin{vmatrix} 1 & s_1 & 0 & \cdots & 0 & s_1^2 \\ 1 & t_1 & 0 & \cdots & 0 & t_1^2 \\ 1 & 0 & t_2 & \ddots & \vdots & t_2^2 \\ \vdots & \vdots & \ddots & \ddots & 0 & \vdots \\ 1 & 0 & \cdots & 0 & t_d & t_d^2 \\ 1 & t & \cdots & t & t & dt^2 \end{vmatrix}_{d+2},\ S_2 = \begin{vmatrix} 1 & t & t & \cdots & t & dt^2 \\ 1 & t_1 & 0 & \cdots & 0 & t_1^2 \\ 1 & 0 & t_2 & \ddots & \vdots & t_2^2 \\ \vdots & \vdots & \ddots & \ddots & 0 & \vdots \\ 1 & 0 & \cdots & 0 & t_d & t_d^2 \\ 1 & s & \cdots & s & s & ds^2 \end{vmatrix}_{d+2}. \tag{2.12}$$

Now we want to derive a general determinant formula for DB–matrices.

Theorem 9 *Let A be a DB–matrix of order n with generating functions c_1, c_2, r_1, r_2, d, then the following identity holds:*

$$
\begin{aligned}
|A| = {} & c_2(n)\textsc{Arrow}(n-1, r_{1|_{1..n-1}}, c_{1|_{1..n-1}}, d_{|_{1..n-1}}) \\
& - c_1(n)\textsc{Arrow}(n-1, r_{1|^{1=c_2(1)}_{1..n-1}}, c_{2|_{1..n-1}}, d_{|^{1=c_2(1)}_{1..n-1}}) \\
& - \sum_{l=2}^{n-1} r_2(l)\textsc{Rform}(n-1, c_{1|^{l\to n-1}_{n-1}}, c_{2|^{l\to n-1}_{n-1}}, r_{1|^{l}_{n-1}}, d_{|^{l}_{n-1}})
\end{aligned}
\tag{2.13}
$$

where $f_{|_{1..m}}$ as before and

$$
f_{|^{k}_{m}}(i) = \begin{cases} f(i) & 1 \le i < k \\ f(i+1) & k \le i \le m \end{cases}
$$

and

$$
f_{|^{k\to m}_{m}}(i) = \begin{cases} f(i) & 1 \le i < k \\ f(i+1) & k \le i < m \\ f(k) & i = m. \end{cases}
$$

Proof.

We will prove the identity by expansion of the last row:

$$
|A| = (-1)^{n+1}c_1(n)|A_{n1}| + \sum_{l=2}^{n-1}(-1)^{n+l}r_2(l)|A_{nl}| + c_2(n)|A_{nn}|.
$$

We observe analogously to the previous theorem that $|A_{n1}| = (-1)^{n-2}\textsc{Arrow}(n-1, r_{1|^{1=c_2(1)}_{1..n-1}}, c_{2|_{1..n-1}}, d_{|^{1=c_2(1)}_{1..n-1}})$ and $|A_{nn}| = \textsc{Arrow}(n-1, r_{1|_{1..n-1}}, c_{1|_{1..n-1}}, d_{|_{1..n-1}})$.

Let $2 \le l \le n-1$, then

$$
|A_{nl}| = \left|\begin{array}{cccccccc}
c_1(1) & r_1(2) & \cdots & r_1(l-1) & r_1(l+1) & \cdots & r_1(n-1) & c_2(1) \\
c_1(2) & d(2) & 0 & \cdots & 0 & \cdots & 0 & c_2(2) \\
\vdots & 0 & \ddots & \ddots & \vdots & & \vdots & \vdots \\
c_1(l-1) & \vdots & \ddots & d(l-1) & 0 & \cdots & 0 & c_2(l-1) \\
c_1(l) & 0 & \cdots & 0 & 0 & \cdots & 0 & c_2(l) \\
c_1(l+1) & 0 & \cdots & 0 & d(l+1) & \ddots & \vdots & c_2(l+1) \\
\vdots & \vdots & & \vdots & \ddots & \ddots & 0 & \vdots \\
c_1(n-1 & 0 & \cdots & 0 & \cdots & 0 & d(n-1) & c_2(n-1)
\end{array}\right| .
$$

Swapping row l down to the bottom, we get

$$|A_{nl}| = (-1)^{n-l-1} \begin{vmatrix} c_1(1) & r_1(2) & \cdots & r_1(l-1) & r_1(l+1) & \cdots & r_1(n-1) & c_2(1) \\ c_1(2) & d(2) & 0 & \cdots & \cdots & \cdots & 0 & c_2(2) \\ \vdots & 0 & \ddots & \ddots & & & \vdots & \vdots \\ c_1(l-1) & \vdots & \ddots & d(l-1) & \ddots & & \vdots & c_2(l-1) \\ c_1(l+1) & \vdots & & \ddots & d(l+1) & \ddots & \vdots & c_2(l+1) \\ \vdots & \vdots & & & \ddots & \ddots & 0 & \vdots \\ c_1(n-1) & \vdots & & & & \ddots & d(n-1) & c_2(n-1) \\ c_1(l) & 0 & \cdots & \cdots & \cdots & \cdots & 0 & c_2(l) \end{vmatrix}$$

which is exactly the R–determinant stated in the theorem. The signfactor reduces to -1, thus we have proved our claim.

$\Box$

Note that the proof is not limited to expansion of the last row. Similar results are obtained if we expand the first row or one of the bordering columns with the effect that we have to deal with oriented arrow and R–forms.

For the sake of completeness we state a corresponding result for DB–matrices with a nonzero counter main diagonal instead of nonzero main diagonal.

Corollary 8 *The determinant of a matrix A of order n with generating functions c_1, c_2, r_1, r_2 like in normal DB–matrices and modified diagonal generating function $\hat{d}(i) = a_{n-i+1,i}$ is*

$$|A| = (-1)^{\lfloor \frac{n}{2} \rfloor - 1} \text{DB}\textsc{form}(n, c_1, c_2, r_1^R, r_2^R, \hat{d}^R),$$

where $r_1^R(i) = r_1(n-i+1), r_2^R(i) = r_2(n-i+1)$ and $\hat{d}^R(i) = d(n-i+1)$.

Proof.

Just swap column j and column $n-j+1$ for $j = 2, \ldots, \lfloor \frac{n}{2} \rfloor - 1$.

$\Box$

Observation

We can express the determinant of a DB–matrix by $2n-2$ determinants of arrow–matrices of smaller order. However, in general, we do not have an explicit determinant formula for symbolic n. It is possible to receive an explicit formula for special cases:

If one of the bordering rows or columns contains only a fixed number (i.e. independent of n) of nonzero entries then a fixed number of determinants of arrow–matrices is sufficient to express the DB–determinant, hence we may obtain an explicit formula for the DB–form. We also observe that we may yield the previous case if two rows or columns are "almost" linear dependent (that is apart form a fixed number of entries) by subtracting an appropriate multiple.

Of course we want to apply our formula to the in–sphere determinants of the previous example and compare them to the results stated in [ES95].

Consider S_1 in (2.12), if we use our package function `DBform` which implements the general formula of Theorem 9, we get the formula

$$\begin{aligned}&(1-s_1/t_1)\cdot\left(dt^2\cdot\prod_{l=1}^{d}t_l\ -t\cdot\prod_{l=1}^{d}t_l\ \sum_{l=1}^{d}t_l\right)\\ &-(s_1^2-t_1s_1)\cdot\left(\prod_{l=1}^{d}t_l\ -t\prod_{l=1}^{d}t_l\ \sum_{l=1}^{d}\frac{1}{t_l}\right).\end{aligned}$$

This formula can be simplified removing the nonzero factor $(s_1/t_1-1)\cdot\prod_{l=1}^{d}t_l$ and we receive

$$t_1+\cdots+t_d-dt+t_1s_1\left(\frac{1}{t_1}+\cdots+\frac{1}{t_d}-\frac{1}{t}\right) \tag{2.14}$$

as in [ES95].

Now we turn to S_2 in (2.12). Using our package function we get the formula

$$\begin{aligned}(1\quad-\quad t/s)\cdot&\left(ds^2\cdot\prod_{l=1}^{d}t_l\ -s\cdot\prod_{l=1}^{d}t_l\ \sum_{l=1}^{d}t_l\right)\\ -\quad(dt^2-dst)\cdot&\left(\prod_{l=1}^{d}t_l\ -s\cdot\prod_{l=1}^{d}t_l\ \sum_{l=1}^{d}\frac{1}{t_l}\right).\end{aligned}$$

It is possible to factor out the nonzero term $(\prod_{l=1}^{d}t_l)/((t-s))$, receiving

$$t_1+\cdots+t_d-dt+dst\left(\frac{1}{t_1}+\cdots+\frac{1}{t_d}-\frac{1}{t}\right) \tag{2.15}$$

as in [ES95]. Both determinants are nonzero for the given ranges of the entries (see [ES95]).

Let us play around with the diagonal a little bit again. As in the previous sections it is obvious that moving the diagonal beyond the first upper or lower side diagonal results in a zero determinant.

The case of nonzero first upper or lower side diagonal is quite interesting however, since it yields only two R–form calls: If we expand the row or column with only two elements, we are left with a minor in R–form and another minor which can easily be transformed into R–form. This gives us the following corollary.

Corollary 9 *Let A be a DB–form of order n with column generating functions c_1, c_2, row generating functions r_1, r_2 and modified function $d(i) = a_{i,i+1}$, $i = 1, \ldots, n-1$ generating the first upper side diagonal, then*

$$\begin{aligned}|A| = (-1)^n(\quad r_2(2)\quad &\textsc{Rform}(n-1, c_{1|_{1..n-1}}, c_{2|_{1..n-1}}, r_{1|^{2}_{n}}, d)\\ &-r_1(2)\textsc{Rform}(n-1, c_{1|^{1=c_1(n)}_{1..n-1}}, c_{2|^{1=c_2 n)}_{1..n-1}}, r_{2|^{2}_{n}}, d_{|^{1=c_1(n)}_{1..n-1}})).\end{aligned}$$

Having the first lower side diagonal instead of the first upper side diagonal in the corollary yields an analogous formula.

Note that in general we cannot obtain a formula for DB–forms with two neighbouring nonzero diagonals since they cannot be reduced to a sum over R–forms.

2.6 Transformation into border form

So far, our matrices had only bordering rows and columns nonzero which we will denote as *border form*. We will investigate the case that the position of the (at most two) rows and columns is arbitrary which we will denote as *frame form*. Of course, we would like that matrices of this more general class can be transformed into border form which would mean a generalization of our preceding results.

In the following we restrict ourselves to determinants of matrices with only the maindiagonal and two bordering rows and columns nonzero. W.l.o.g. we assume that there are two nonzero rows and two nonzero columns which are not at bordering positions.

Having a matrix of order n of this form, we proceed as follows:

1. Swap the first nonzero row at position k_1 with the first row (and consider the effect on columns and diagonal). The first diagonal element a_{11} is now at $a_{k_1 1}$ and if the first column is not nonzero we have to get rid of this "dangling" element to preserve the form with only one diagonal, two rows and two columns nonzero. We swap the first column with the k_1th column which gets the "dangling" element back into the diagonal (effects on the rows and diagonal have to be considered) and restored our desired form since the element $a_{k_1 k_1}$ is zero except if column k_1 is nonzero but in this case we would swap the column into the right place.

2. Swap the second nonzero row at position k_2 with the last row (effects on columns and diagonal have to be considered). If the last column is zero with a "dangling" element then we have to swap column n with column k_2. Now the rows are at the desired place.

3. Swap the first nonzero column with the first column (considering the effect on rows and diagonal). Since the rows are already at bordering positions, we don't have to get rid of an unwanted "dangling" element.

4. Swap the second nonzero column with the last column (considering the effect on rows and diagonal). Finally, we reached the border form. The determinant of the transformed matrix multiplied with the sign factor introduced by the swapping is identical to the original matrix.

Let us illustrate the process in an example:

Example

Consider the determinant

$$\begin{vmatrix} a & 0 & b & 0 & c & 0 & 0 & 0 \\ d & a & b & d & c & d & d & d \\ 0 & 0 & a & 0 & c & 0 & 0 & 0 \\ 0 & 0 & b & a & c & 0 & 0 & 0 \\ e & e & b & e & a & e & e & e \\ 0 & 0 & b & 0 & c & a & 0 & 0 \\ 0 & 0 & b & 0 & c & 0 & a & 0 \\ 0 & 0 & b & 0 & c & 0 & 0 & a \end{vmatrix}.$$

Swapping row 2 with row 1 gets one of the rows into the desired place but introduces a dangling element:

$$-\begin{vmatrix} d & a & b & d & c & d & d & d \\ a & 0 & b & 0 & c & 0 & 0 & 0 \\ 0 & 0 & a & 0 & c & 0 & 0 & 0 \\ 0 & 0 & b & a & c & 0 & 0 & 0 \\ e & e & b & e & a & e & e & e \\ 0 & 0 & b & 0 & c & a & 0 & 0 \\ 0 & 0 & b & 0 & c & 0 & a & 0 \\ 0 & 0 & b & 0 & c & 0 & 0 & a \end{vmatrix}.$$

Successive exchange of the first two columns eliminates the dangling element and yields

$$\begin{vmatrix} a & d & b & d & c & d & d & d \\ 0 & a & b & 0 & c & 0 & 0 & 0 \\ 0 & 0 & a & 0 & c & 0 & 0 & 0 \\ 0 & 0 & b & a & c & 0 & 0 & 0 \\ e & e & b & e & a & e & e & e \\ 0 & 0 & b & 0 & c & a & 0 & 0 \\ 0 & 0 & b & 0 & c & 0 & a & 0 \\ 0 & 0 & b & 0 & c & 0 & 0 & a \end{vmatrix}.$$

Next, we swap row 5 with row 8 and then column 5 and 8 to get rid of the resulting dangling element and get

$$\begin{vmatrix} a & d & b & d & d & d & d & c \\ 0 & a & b & 0 & 0 & 0 & 0 & c \\ 0 & 0 & a & 0 & 0 & 0 & 0 & c \\ 0 & 0 & b & a & 0 & 0 & 0 & c \\ 0 & 0 & b & 0 & a & 0 & 0 & c \\ 0 & 0 & b & 0 & 0 & a & 0 & c \\ 0 & 0 & b & 0 & 0 & 0 & a & c \\ e & e & e & e & e & e & e & a \end{vmatrix}.$$

The last transformation already put one of the columns into the right position, thus, it remains to swap column 3 with column 1 and we finally get

$$-\begin{vmatrix} b & d & a & d & d & d & d & c \\ b & a & 0 & 0 & 0 & 0 & 0 & c \\ a & 0 & 0 & 0 & 0 & 0 & 0 & c \\ b & 0 & 0 & a & 0 & 0 & 0 & c \\ b & 0 & 0 & 0 & a & 0 & 0 & c \\ b & 0 & 0 & 0 & 0 & a & 0 & c \\ b & 0 & 0 & 0 & 0 & 0 & a & c \\ b & e & e & e & e & e & e & a \end{vmatrix},$$

which is in the desired frame form.

How does the procedure change if we take one of the principal side diagonals to be nonzero instead?

Essentially, we follow the four steps above, with the difference that the "dangling" element appears only in the second or penultimate row or column and has to be treated appropriately. In half of the swap cases, there are no dangling elements (e.g. if the lower side diagonal is nonzero and we want to swap row $k_1$and the first row then no "cosmetics" are needed afterwards).

However, there are a few special cases to be taken into account: Consider the case that the upper side diagonal, an arbitrary row and the first and the second column are nonzero. Assume that the row has already been transformed without changing the position of the columns:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} & a_{14} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & 0 & \cdots & 0 \\ a_{31} & a_{32} & 0 & a_{34} & \ddots & \vdots \\ a_{41} & a_{42} & 0 & 0 & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & a_{n-1,n} \\ a_{n1} & a_{n2} & 0 & 0 & \cdots & 0 \end{vmatrix} .$$

Since the first column is already in place we have to swap the second column with the last column which in general introduces a dangling element at the position $(n-1,2)$. Unfortunately, now, it's not possible to get rid of this element by performing another row or column exchange without introducing an additional row or column with nonzero elements. However, we can escape the dilemma if we simply swap the second column at position two through to position n performing $n-2$ column exchanges. The resulting matrix then has a nonzero main diagonal :

$$(-1)^{n-2} \begin{vmatrix} a_{11} & a_{13} & a_{14} & \cdots & a_{1n} & a_{12} \\ a_{21} & a_{23} & 0 & \cdots & 0 & a_{22} \\ a_{31} & 0 & a_{34} & \ddots & \vdots & a_{32} \\ \vdots & \vdots & \ddots & \ddots & 0 & \vdots \\ a_{n-1,1} & 0 & \cdots & 0 & a_{n-1,n} & a_{n-1,2} \\ a_{n1} & 0 & \cdots & 0 & 0 & a_{n2} \end{vmatrix} .$$

We proceed analogously in the case of counter diagonals.

Observe that we can easily drop the assumptions having exactely two nonzero rows and columns in non–bordering positions at the beginning of our procedure.

It is important to note that such transformations are not possible in general if we allow two (or more) nonzero diagonals:

Example

Consider the following determinant:

$$\begin{vmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix} .$$

Exchanging row one and four yields

$$-\begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}.$$

Now, we introduced two "dangling" elements. We can get rid of the first one in a_{41} by simply exchanging column one and four. The second one causes problems, however. We cannot get rid of this dangling element without causing another.

Of course, if we would allow that an additional bordering column can be nonzero, we could also deal with this case, but if we already had two nonzero columns, we are definitely lost.

In many special cases, a transformation would be possible, but, having automation in mind, we are only interested in a general solution which can only be found in the case of one nonzero diagonal.

It may be noted that our transformations are based on graph isomorphism if we use an appropriate representation of the frame form determinants.

2.7 Implementation

We address implementation issues of the Maple package FRAMEFORMS that provides the computation of determinant formulas for specified matrices of the discussed shapes. Examples and further details can be found in the appendix or the on–line help pages.

2.7.1 General Considerations

Which matrix generating functions can I use ? In the previous sections we defined the matrix generating functions in a general way: A generating function $f(i), i = 1, \ldots n$ could take any values for any fixed i, that is there are arbitrary piecewise definitions possible.

In our implementation we will need to restrict this generality to make the coding bearable.

We now require that piecewise definitions of f are only possible in the intervals $[1..p_1]$ and $[n - p_2..n]$ with integer p_1, p_2. In these intervals we may choose any value for $f(i)$ but in the interval $[p_1 + 1..n - p_2 - 1]$ we need $f(i) = g(i)$ for a non–piecewise function g.

This restriction is necessary since any slighter generalization amounts in an unacceptable amount of additional coding.

However, many interesting examples can be handled using this restriction. Another reason is that the determinant formulas will get more and more complicated and unreadable as the generating functions become more and more general.

Consequently, we restrict the matrix order n to be of the form $n = v + d$ with a symbolic variable v and integer d. This is reasonable considering the first restriction.

Matrix order The matrix order n can either be a positive integer or a symbolic value (see above). The first case results in a determinant which could also be computed by Maple's `det` function. For small values of n this will be significantly faster due to all the testing overhead in our functions. However, for very large values of n it may be worth to use the package, since it provides direct computation without dealing with storage consuming intermediate expressions.

Specifying the matrix This is the most important issue: How can we specify matrices of this class?

The idea is very intuitive. If we have a special matrix in mind or on paper and should specify it, we would say for example: “the first column looks like this, the last row looks like this and the main diagonal like this”.

Thus the matrix specification is a list of row, column or diagonal specifications:

`specM=[ specL_1, ... , specL_k ]`

In this list we have lists of tuples of the form `specL = [ type[pos] , functionspecL ]` specifying a certain matrix chunk.

`type` may be `row`,`col`,`diag` or `cdiag`.

`pos` can be between `1` or `n` for `row` and `col` type or `-1`,0,1 for `diag` and `cdiag` type.

Obviously, `col[n]` means the last column and `diag[-1]` means the first lower side diagonal.

`functionspecL` is the piecewise specification of the elements in the current matrix chunk.

`functionspecL= [ [interval_1, function_1(i)], ... , [interval_k,function_k(i)]]`

Example

The following matrix

$$M = \begin{pmatrix} a_1 & a_2 & \cdots & a_{n-1} & a_n \\ -1 & x & 0 & \cdots & 0 \\ 0 & -1 & x & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & x \end{pmatrix}$$

would be specified by

`[[row[1],[[1..n,a[i]]] ], [diag[0],[[1..1,a[1]],[2..n,x]] ],[diag[-1],[[1..n-1,-1]] ] ]`

Features:

- If an entire row, column or diagonal is generated by a single function `f(i)` , it is cumbersome to always write `[[1..n,f(i)]]`. This can also be achieved with `f(i)` or `[1..n,f(i)]` for short.
- Since corner elements overlap it is also cumbersome, that we have to to write this corner element in every relevant function list. Hence we allow that it may be left out if already specified elsewhere. A shorter specification for the preceding example would therefore be: `[ [row[1],a[i] ] , [ diag[0], [2..n,x] ] , [diag[-1],-1] ]`
- It is also possible to leave holes at the beginning and the end of a function specification list which are padded with zeroes.

  I.e. `[[4..7,a],[8..n-3,b]]` is modified into `[[1..3,0],[4..7,a],[8..n-3,b],[n-2..n,0]]` if the corner elements were not specified yet, otherwise it is adjusted appropriately.
- Another short cut is `diag` for `diag[0]` and `cdiag` for `cdiag[0]` respectively.
- If an interval only contains one element we may write `[p,f(i)]` instead of `[p..p,f(i)]`.

Note that the single specifications of rows, columns and diagonals can be in arbitrary order.

If no specification for overlapping corner elements is found, they will be treated as zero.

Checking the determinant How can we trust the computation of a determinant formula using this package? It would be desirable if we could test the resulting determinant formula with different values for the matrix order `n`.

Hence, if we give the optional directive "check[dim]", the positive integer `dim` is substituted for symbolic order n into the matrix specification and the determinant of the resulting matrix of order `dim` is computed using Maple's `det` function and compared with the output formula where we substituted `dim` for `n` as well. Inconsistencies are reported to the user as a sign that the output formula is not trust worthy.

Setting `dim` to a very large value, or successively trying different check values, increases the trust in the result. On the other hand, large values for `dim` also increase computation times and can fail when Maple runs out of memory at some stage.

The default check value is 4. It is automatically adjusted to c if the output formula is only valid for $n \geq c$.

2.7.2 The package functions

The `FRAMEFORMS` package consists of the following functions:

`Frameform, FrameformMatrix, GetBorderForm, Form7, Arrow, Nform, Rform, DBform.`

The function `Frameform` automatically detects the matrix form and calls the corresponding function. The function `FrameformMatrix` simply returns the specified matrix and serves as a tool to play around with (e.g. one might be interested in the product of such matrices). In the case of symbolic orders we use dots "o" to abbreviate the symbolic order appropriately and hence the returned matrix should just be used for illustrative purposes since the "dots" are treated as normal entries by Maple. The function `GetBorderForm` returns the specified frame form matrix in border form performing the operations described in the last section.

2.7.2.1 General Properties

The general structure of the package functions could be loosely described as:

1. Parse and test input determining additional informations and transforming the specified matrix into standard border form if necessary.
2. Transform specified matrix into standard form to apply our derived formula.
3. Compute the determinant formula according to number and position of the diagonals splitting up the occurring sums (and products) into the necessary parts, simplify and test (if specified).

All package functions first parse and test the input. They all call the function `MatrixSpecification` which checks the matrix order `n` and the matrix specification `specL` according to the previous section. If the optional argument `print` is given it calls the function `printMatrix` for integer matrix orders or `printMatrixSymbolic` in the symbolic case. These functions display the specified matrix using dots to illustrate the symbolic case.

The optional argument `check` or `check[k]` enables the checking mechanism. The check order is determined (see above) and the internal function `printMatrix` is called for that order with the option not to print but just returning the appropriate matrix . Maple's `det` function computes the trustworthy check result.

If the specified matrix is not already in border form, the function `determineStandardFRAMEFORM` is called which transforms the lists of the generating functions appropriately (according to the previous section taking all special cases into account) such that they would result in the specification of the corresponding transformed matrix. A possible sign factor is also returned.

The final stage of `MatrixSpecification` is to determine the type and corresponding orientation of the specified matrix.

It eventually returns a list of informations: `[ Type,n, V, symbolic, checkresult, checkdim, ndiags, diagpos, orient, L]` .

- `Type` is one of the unevaluated names `diagonal, arrow, form7, N, R, DB` .
- `V` is the variable in the matrix order (if `n` is symbolic, 0 otherwise).
- `symbolic` is a boolean flag describing the matrix order.
- `check_dim` is the matrix order used for checking the result (0 if checking is disabled).
- `check_result` is the determinant of order `check_dim` computed with Maple's `det` function.
- `n_diags` is the number of occurring diagonals.
- `diag_pos` is the position of the "highest" or "lowest" diagonal (0 for main diagonal, 1 for upper side diagonal, -1 for lower side diagonal).
- `orient` is the orientation of the matrix form. This indicates how to transform into standard form.
- `L` is a sequence of function specification lists depending on `Type` e.g. for arrow type we return the function list specifying the row, followed by the function list s specifying the column and the diagonal(s).

The package functions then test whether the determined type is correct and call their computation functions with the additional informations .

The computation functions first transform their input matrix expressed by the generating function lists to standard form (to make use of our formulas). We need to manipulate the function lists for that purpose using package internal list modification functions. Transposing the matrix and swapping row (or column) pairs $k, n-k+1$, for $k = 1, \dots, \lfloor \frac{n}{2} \rfloor$ are sufficient for these transformations. The internal package function `reverseL` simulates the effect of pairwise swapping on the function lists. Sign factors arising during the transformations are stored separately and multiplied to the final result formula to keep intermediate results more readable.

After having reached the standard form we are able to compute the appropriate determinant formula (according to the number of diagonals and their position). The remaining problem is to split up the sums and products in the formula. This is quite easy for integer determinant orders, whereas the case of symbolic determinant orders has to be examined more closely: We allowed the defining functions to be piecewise in the intervals `[1..p1]`, `[n-p2+1..n]` (p1,p2, integer) and required a total generating function in [p1+1..n-p2], an interval with length dependent on n. Therefore we compute the maximum p1 and p2 of all involved defining functions and compute $\sum_{l=1}^{p1} formula + \sum_{l=p1+1}^{n-p2} formula + \sum_{l=n-p2+1}^{n} formula$. The second sum is over an interval which is dependent on the symbolic variable n, hence we will make use of Maple's `sum` command which is able to find closed forms for many sums. To avoid name clashes with the input we always use underscored names for summation or product indices.

If the formula involves computation of a determinant of another form we have to manipulate the function lists appropriately to get the correct new defining functions. This is done using the internal list modification functions like `chopFirst, chopLast, addFirst, addLast, move_i_to_j`.

In the end we apply Maple's `simplify` function on the resulting formula with the sign factor. If checking is enabled we call the function `checkResult` which compares the correct check result with the formula's value for `n=check_dim`.

2.7.2.2 Special Properties

Following we discuss special implementation issues of the different functions.

Form7 The only difficulty here after the transformation into standard form is to handle the piecewise function definitions. After splitting up the sum we also have to worry about the products which also need to be split up.

Arrow We distinguish the case of only one diagonal and two diagonals.

An Arrow form with one diagonal is transformed into standard form. Simple formulas are computed for forms that have the upper or lower main diagonal nonzero. In the main diagonal case we determine the number of zeroes of the diagonal function for $i = 2, \ldots, n$. If this number is ≥ 2 we have a zero determinant if the maindiagonal vanishes for $i = zeropos$ the determinant reduces to the term $-r(i)c(i)\prod_{\substack{l=2 \\ l \neq zeropos}}^{n} d(l)$. Otherwise we compute $D = \prod_{l=2}^{n} d(l)$ and can safely compute the formula $d(1)D - D\sum_{l=2}^{n} \frac{r(l)c(l)}{d(l)}$.

The two diagonal case is somewhat tricky: First we transform the matrix into standard form with main diagonal and upper side diagonal nonzero. The formula (2.8) involves the determinant of a 7–form matrix of order l in the sum. Due to the possibility of piecewise function definition it is not sufficient to compute the formula of this 7–form of order n and substituting n appropriately. We divide the sum again into three parts. The fixed order 7–forms in the first sum are computed constructing the 7–form of order p_1 and using Maple's `det` for the determinant of this matrix and its minors. In the second sum we can safely use substitution in the formula of the 7-matrix of order $n - p_2 - 1$. The third sum poses problems again. We compute the 7–form formula for all necessary values and use substitution for the others.

Our decision to restrict the piecewise function definitions will become obvious here at the latest.

Nform The implementation of N–forms is straightforward. First we test whether the rows or the columns constituting the N–form are linearly dependent which would result in a zero determinant. In the case of only one diagonal we transform the input to standard N–form and compute the resulting formula distinguishing between the position of the diagonal.

In the two diagonal case we transform the input to a N–form with nonzero columns, main diagonal and lower main diagonal. Choosing this transformation yields standard 7–form matrices expanding the last row which eliminates additional transformations.

Rform Preceding again is the linear dependency test followed by the transformation to standard R–form. The case of one diagonal results in different computations with two Arrow form invocations each. The two diagonal case with nonzero first upper side diagonal yields similar Arrow form function calls (this time with "fat shaft"). The two diagonal case with nonzero first lower side diagonal is treated specially since we only determined a recurrence formula in Corollary 7: We simulate the stepwise elimination of the lower diagonal with appropriate row operations and obtain a standard R–form with complicated column entries that is solved like above (yielding very complicated formulas in general).

DBform Here we only allow one nonzero diagonal.

After the linear dependency checking for the rows and the columns we transform the matrix into standard form.

If the nonzero diagonal is not the maindiagonal we can use the simple expansion of a row consisting of only two nonzero elements which results in only two R–form calls.

Having the main diagonal nonzero we're left with the difficult formula and our aim will be finding or producing as many zeroes as possible in one of the rows and columns.

If none of the bordering rows and columns contain or can be transformed to contain but a fixed (integer) number of nonzero elements then our formula tells us that we are left with a "symbolic" sum of R–form invocations which cannot be evaluated. Otherwise we transform again such that the row or column with the fixed number of nonzero entries is located in the first row. Following we can compute the result using our formula with the advantage of having only a fixed number of R–form calls.

2.7.3 Problems

One of the major drawbacks of the package is that the output determinant formulas are not entirely simplified as one would desire, which leads to quite big formulas for more complicated problems. This is due to the fact that we did not implement simplification strategies for special cases of the discussed determinant forms and due to Maple's inability to recognize some potential simplification patterns in expressions . Although Maple is designed to allow users to define their own simplification procedures this would go beyond the scope in our case. So we have to look ourselves if we find any obvious simplifications to the formulas. It is suggested to reapply Maple's `simplify` function or to try the `factor` function which sometimes (but not always) produces a more readable formula.

Unfortunately, there seems to be a bug in Maple's `sum` function, hence our package functions don't always work properly (even after the imposed restrictions) in the case of two diagonals. The reasons are Maple's attempts to find closed forms for sums which sometimes result in an internal error configuration or a formula containing ∞ (obtained by limit computation) that cannot be checked for integer orders.

These errors occur when the defining matrix functions become more and more complex, especially when more of them are non–constant functions in i which often means that Maple attempts to find a closed form for sums over non–trivial products.

A simple avoidance of these errors would be using Maple's command for inert summation, `Sum` which returns the unevaluated sum. However, in the simpler cases we are happy about closed forms of the occurring sums, in fact that's what is expected from such a package.

We chose to steer a middle course: After the in `sum` call in `Form7` we check whether the result contains ∞ and replace it by the inert `Sum` call in that case. This reduces the number of errors significantly without giving up the advantage of getting nice closed forms for some of the sums.

2.8 Summary

In this chapter we derived a number of determinant formulas for matrix classes that only had bordering rows and columns as well as the main diagonal nonzero. We generalized the results for arbitrary position of the rows and columns. The effect of an additional neighbouring diagonal was examined, obtaining formulas in most cases apart from the DB–form.

The designed Maple package FRAMEFORMS allows the (somewhat restricted) specification of a frame form matrix and computes a determinant formula using the results of this chapter. The package can be used to derive determinant formulas for some special geometric predicates in determinant form like in [ES95]. However, the resulting formulas tend to be in need of further manual simplification. "Nice" formulas are only obtained in simple cases.

We will often meet determinants of the discussed forms in later chapters, e.g. investigating alternants and double alternants or symmetric determinants.

Chapter 3

Alternants

In this chapter we are investigating determinants of a very important matrix class, so–called alternants. They occur in various applications, such as interpolation and geometric primitives.

We will first examine general properties of these alternants, illustrate different methods to compute their determinant exploiting their relation to elementary and complete symmetric functions. Then we try to generalize our results to double–alternants.

3.1 Alternants

3.1.1 Definition and basic properties

Let us define the term of an *alternant*. We will first give an illustrative definition followed by an extended definition.

Definition 12 *Let A be a matrix of order n. If the entries of the first row are generated by functions $f_1, \ldots, f_n$ in one variable x_1, the entries of the second row by the same functions in another variable x_2 and so on then $|A|$ is called an* alternant.

$$|A| = \begin{vmatrix} f_1(x_1) & f_2(x_1) & \cdots & f_n(x_1) \\ f_1(x_2) & f_2(x_2) & \cdots & f_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(x_n) & f_2(x_n) & \cdots & f_n(x_n) \end{vmatrix}$$

Obviously, we would also have an alternant if the columns would be generated by the functions in a certain variable since transposing leaves the determinant unaltered and reduces it to the previous case.

Let us try to explain the name alternant: Every alternant of order n is a function of n variables. To exchange two of these would be the same as to exchange two rows (or columns), and therefore would have the effect of merely changing the sign of the function. Since functions with this property are known as *alternating functions*, the origin of the name alternant is apparent.

Therefore we have the following extended definition:

Definition 13 *Any determinant which is an alternating function is called an alternant.*

Notation We will often denote an alternant $|A|$ with generating functions $f_1, \ldots, f_n$ in the variables $x_1, \ldots, x_n$ by its main diagonal elements: $|A(f_1(x_1), f_2(x_2), \ldots, f_n(x_n))|$

We present the most popular and very important example:

Example

The *Vandermonde matrix* V of order n is defined as $v_{ij} = x_i^{j-1}$. Its determinant

$$|V| = \begin{vmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1} \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{vmatrix}$$

is an alternant of order n with column generating functions x_i^{j-1} , $j = 1, \ldots, n$.

Vandermonde matrices have important applications. Recall the problem of polynomial interpolation:

Given $n+1$ pairs of points $(x_0, f_0), (x_1, f_1), \ldots, (x_n, f_n)$ with distinct x_i, we want the unique polynomial $p(x) = \sum_{j=0}^{n} a_j x^{j-1}$ that interpolates the values f_i in the points x_i, that is, $p(x_i) = f_i$. Writing this as a linear system in matrix form we get:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^n \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \\ a_n \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_{n-1} \\ f_n \end{pmatrix}.$$

Since the x_i are distinct, the Vandermonde matrix of order $n+1$ is nonzero which we will see in the following and the linear system has a unique solution.

After having studied possible applications of Vandermonde determinants, we are interested in the formula of this determinant. Almost every maths course on linear algebra requires from its students to prove that $|V| = \prod_{i<j}(x_j - x_i)$ (usually by induction). We will derive this formula in the following using a different proof method but first we will state a fundamental property of alternants:

Theorem 10 *Every alternant of order n with entries in the ring of polynomials* $R[x_1, \ldots, x_n]$ *contains the* difference product

$$DP(x_1, \ldots, x_n) = \prod_{1 \le i < j \le n}^{n} (x_j - x_i)$$

of its variables $x_1, \ldots, x_n$ *as a factor.*

Proof.

Let the variables be $x_1, \ldots, x_n$.

Substituting x_n for any other of the variables we cause the determinant to vanish since we would obtain two similar rows. Hence it follows that $\prod_{i=1}^{n-1}(x_n - x_i)$ is a factor. Similarly substituting x_{n-1} for the other variables yields a factor $\prod_{i=1}^{n-2}(x_{n-1} - x_i)$ and so on.

□

This is a very important theorem which allows us to make some simplifications to find the cofactor of the difference product as we will see in the following sections.

It remains to prove the Vandermonde determinant identity. In fact, we will prove that the difference product can be expressed as an alternant, namely the Vandermonde determinant.

Theorem 11 *The difference product $DP(x_1, \dots, x_n)$ is expressible as an alternant, namely the Vandermonde alternant $|V|$:*

$$DP(x_1, \dots, x_n) = \begin{vmatrix} 1 & x_1 & \cdots & x_1^{n-1} \\ 1 & x_2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \cdots & x_n^{n-1} \end{vmatrix}.$$

Proof.

Assume the following lexicographic ordering of the variables: $x_1 \preceq x_2 \preceq \ldots \preceq x_n$.

Thus the leading term of $\mathrm{DP}(x_1, \dots, x_n) = \prod_{1 \le i < j \le n}(x_j - x_i)$ is $x_n^{n-1} x_{n-1}^{n-2} \cdots x_2$.

What is the leading term of the Vandermonde alternant? It follows from its structure and the basic definition of determinants that the leading term is the product of the main diagonal elements $x_n^{n-1} x_{n-1}^{n-2} \cdots x_2$.

Since $|V|$ contains $\mathrm{DP}(x_1, \dots, x_n)$ as a factor and their leading factors coincide (including the coefficients), it follows that indeed $\mathrm{DP}(x_1, \dots, x_n) = |V|$.

□

The preceding results allow some straightforward conclusions on certain Vandermonde like alternants. We will denote an alternant of order n with column generating functions $f_1, \dots, f_n$ and variables $x_1, \dots, x_n$ with its main diagonal elements $|f_1(x_1), f_2(x_2), \dots, f_n(x_n)|_n$or briefly $|(f_j(x_i))|_n$ and define $\mathrm{DP}(p(x_1), \dots, p(x_n)) = \prod_{1 \le i < j \le n}(p(x_j) - p(x_i))$ for polynomials $p(x_i)$.

Corollary 10 *Let $q(x_i)$ be a polynomial independent from j and $l(j) = kj + d$ with $k, d \in \mathbb{N}$. then*

$$|q(x_i)(p(x_i))^{l(j)}|_n = \prod_{i=1}^{n} q(x_i)(p(x_i))^{d+k}\, DP(p(x_1)^k, \dots, p(x_n)^k).$$

Proof.

Since $q(x_i)$ is only a multiplicative factor of the alternant entries independent of j, we can remove this factor from every row. Now we we look at $p(x_i)^{l(j)} = p(x_i)^{k(j-1)+d+k} = p(x_i)^{d+k}(p(x_i)^k)^{j-1}$. The first factor may be taken out again and substituting $u_i = p(x_i)^k$ we are left with the normal Vandermonde alternant.

□

If the generating functions of an alternant are of the form $p_j(x_i) = \sum_{l=0}^{k}(a_l x_i^l + b_l x_i^{j+l})$, it is also possible to derive a formula for this type.

Corollary 11 *Let $|A| = |p_j(x_i)|_n$ be an alternant with generating functions $p_j(x_i) = \sum_{l=-d_1}^{d_2}(a_l x_i^l + b_l x_i^{j+l})$, $d_1, d_2 \in \mathbb{N}$ and a_l, b_l coefficients independent from i, j, then*

$$|A| = \sum_{i=1}^{n}(-1)^{i+1} p_1(x_i) \prod_{\substack{k=1 \\ k \ne i}}^{n} \sum_{l=-d_1}^{d_2} b_l x_k^{l+1}(x_k - 1)\, DP(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n).$$

Proof.

We consecutively subtract column k from column $k+1$ for $k = n-1, \ldots, 1$. The constant powers cancel out after this modification and we have

$$|A| = \begin{vmatrix} p_1(x_1) & \sum_{l=-d_1}^{d_2} b_l x_1^{l+1}(x_1-1) & x_1 \sum_{l=-d_1}^{d_2} b_l x_1^{l+1}(x_1-1) & \cdots & x_1^{n-2} \sum_{l=-d_1}^{d_2} b_l x_1^{l+1}(x_1-1) \\ p_1(x_2) & \sum_{l=-d_1}^{d_2} b_l x_2^{l+1}(x_2-1) & x_2 \sum_{l=-d_1}^{d_2} b_l x_2^{l+1}(x_2-1) & \cdots & x_2^{n-2} \sum_{l=-d_1}^{d_2} b_l x_2^{l+1}(x_2-1) \\ \vdots & \vdots & \vdots & & \vdots \\ p_1(x_n) & \sum_{l=-d_1}^{d_2} b_l x_n^{l+1}(x_n-1) & x_n \sum_{l=-d_1}^{d_2} b_l x_n^{l+1}(x_n-1) & \cdots & x_n^{n-2} \sum_{l=-d_1}^{d_2} b_l x_n^{l+1}(x_n-1) \end{vmatrix}.$$

Expanding the first column, we examine the resulting minors $|A_{1i}|$ and see that we can factor out

$$\sum_{l=-d_1}^{d_2} b_l x_k^{l+1}(x_k - 1)$$

in each row and are left with a Vandermonde determinant of order $n-1$ in the variables $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$ which completes the proof.

Note that the proof would not work if the generating functions $p_j(x_i)$ involved two terms x_i^{j+k} and x_i^{2j+d} simultaneously.

$\square$

Example

Let us illustrate the somewhat complicated formula: Consider the alternant $|A| = |p_j(x_i)|_n$ with $p_j(x_i) = 1 + 2x_i^2 + x_i^{j-1} + x_i^j$.

The corollary yields the formula

$$\sum_{i=1}^{n} (-1)^i (2 + x_i + 2x_i^2) \prod_{\substack{k=1 \\ k\neq i}}^{n} (x_k^2 - 1) \, \mathrm{DP}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)).$$

It would be possible to obtain formulas for other special alternant classes of simple structure using these methods but now we will focus on the cofactor of the difference product. The last corollaries did not enlighten us about the general structure of the cofactor.

We will show that this cofactor is a symmetric function in the variables.

Theorem 12 *Let $|A(x_1, \ldots, x_n)|$ be an alternant of order n. Then*

$$|A(x_1, \ldots, x_n)| = S(x_1, \ldots, x_n) DP(x_1, \ldots, x_n)$$

with $S(x_1, \ldots, x_n)$ being a symmetric function in the variables $x_1, \ldots, x_n$.

Proof.

Examine $\frac{|A(x_1,\ldots,x_n)|}{\mathrm{DP}(x_1,\ldots,x_n)}$: If we exchange any two variables, both the nominator and the denominator change sign (being alternants this has the effect of exchanging two columns), hence the quotient remains unaltered, which is the definition of a symmetric function in these variables.

$\square$

We will study this symmetric cofactor in the following sections. At first we will discuss important properties of symmetric functions that aid our investigations.

3.1.2 Symmetric Functions

We will study the basic important properties of symmetric polynomials. Our presentation follows [CLO96] and [Kal].

Let R be a commutative ring with identity and $x_1, \ldots, x_n$ indeterminates over R.

Definition 14 *A polynomial $f \in R[x_1, \ldots, x_n]$ is said to be* symmetric *if it remains unchanged under all permutations of its variables, i.e.*

$$f(x_{\pi(1)}, \ldots, x_{\pi(n)}) = f(x_1, \ldots, x_n)$$

for all possible permutations π of $\{1, \ldots n\}$.

For example, if the variables are x, y, and z, then $x^2 + y^2 + z^2$ and also xyz are symmetric polynomials.

It is easy to see that the set of all such symmetric polynomials is itself a ring, hence we have a subring of $R[x_1, \ldots, x_n]$. By definition, every element of $R[x_1, \ldots, x_n]$ can be expressed as a polynomial in $x_1, \ldots, x_n$, hence $x_1, \ldots, x_n$ generates the ring $R[x_1, \ldots, x_n]$.

Can we find finitely many symmetric polynomials that generate the ring of symmetric polynomials?

Surprisingly perhaps, the answer to this question has been known since at least the late eighteenth century and in a certain form to Newton.

Definition 15 (Elementary symmetric functions) *Given variables $x_1, \ldots, x_n$, we define the* elementary symmetric functions $\sigma_0, \sigma_1, \ldots, \sigma_n \in R[x_1, \ldots, x_n]$ *as*

$$\begin{aligned}
\sigma_0 &= 1 \\
\sigma_1 &= x_1 + \cdots + x_n, \\
\sigma_2 &= \sum_{1 \le i_1 < i_2 \le n} x_{i_1} x_{i_2}, \\
&\vdots \\
\sigma_r &= \sum_{1 \le i_1 < \cdots < i_r \le n} x_{i_1} \cdots x_{i_r}, \\
&\vdots \\
\sigma_n &= x_1 x_2 \cdots x_n.
\end{aligned}$$

Thus σ_r is the sum of all monomials that are products of r distinct variables. In particular, every term of σ_r has total degree r.

We would like to convince ourselves that these polynomials are indeed symmetric. We introduce a new variable X and consider the polynomial

$$f(X) = (X - x_1)(X - x_2) \cdots (X - x_n) \tag{3.1}$$

with roots $x_1, \dots, x_n$. If we expand the right–hand side, it is straightforward to show that

$$f(X) = X^n - \sigma_1 X^{n-1} + \sigma_2 X^{n-2} + \cdots + (-1)^{n-1}\sigma_{n-1}X + (-1)^n \sigma_n$$

Now suppose that we rearrange $x_1, \dots, x_n$. This changes the order of the factors on the right–hand side of (3.1), but f itself remains unaltered. Thus, the coefficients $(-1)^r\sigma_r$ of f are symmetric functions.

We can conclude that for any polynomial with leading coefficient 1, the other coefficients are the elementary symmetric functions of its roots (up to a factor of ± 1).

From the elementary symmetric functions, we can construct other symmetric functions by taking polynomials in $\sigma_1, \dots, \sigma_n$, in fact what is more surprising is that *all* symmetric polynomials can be represented in this way.

Theorem 13 (Fundamental Theorem of Symmetric Polynomials) *Every symmetric polynomial in $R[x_1, \dots, x_n]$ can be written uniquely as a polynomial in the elementary symmetric functions $\sigma_0, \sigma_1, \dots, \sigma_n$.*

Proof. [Gauß]

See [CLO96] for a proof.

$\square$

At the end of this brief introduction to symmetric functions, we give a definition of the *complete symmetric functions* and state some important identities.

Definition 16 *The* complete symmetric function *of the variables $x_1, \dots, x_n$ of degree m, denoted by $H_m(x_1, \dots, x_n)$ or briefly H_m, is the sum of all possible powers and products of $x_1, \dots, x_n$ of the mth degree. It may be recursively defined as*

$$H_m(x_1, \dots, x_n) = x_n H_{m-1}(x_1, \dots, x_n) + H_m(x_1, \dots, x_{n-1}) \tag{3.2}$$

with $H_m() = 0$ and $H_0 = 1$ and $H_{-k} = 0$ for all variables and $k \in \mathbb{N}$.

The number of terms in $H_m(x_1, \dots, x_n)$ is $\binom{m+n-1}{n-1}$.

It is straightforward to see that the complete symmetric functions also form a basis of the symmetric polynomials. Other bases are the power sum symmetric functions or the Schur functions, both of which we do not use here. It is possible to express any symmetric polynomial in terms of one of these bases and it is possible to convert from one basis to another (There is a Maple package `SF` in the `share` library which provides this conversion for non–symbolic degree).

Let us state some identities about complete symmetric functions that will be useful in the following:

Developing the first term on the right in (3.2) we get:

$$H_m(x_1, \dots, x_n) = x_n^m + x_n^{m-1}H_1(x_1, \dots, x_{n-1}) + x_n^{m-2}H_2(x_1, \dots, x_{n-1}) + \cdots + H_m(x_1, \dots, x_{n-1}). \tag{3.3}$$

Similarly, the development of the second term yields:

$$H_m(x_1, \dots, x_n) = x_n H_{m-1}(x_1, \dots, x_n) + x_{n-1}H_{m-1}(x_1, \dots, x_{n-1}) + \cdots + x_2 H_{m-1}(x_1, x_2) + x_1^m.$$

Clever development and some modifications give us the next identities (see [Met60] p. 332 for details):

$$\begin{aligned} H_m(x_1,\dots,x_n) &= x_1^m + H_1(x_2,\dots,x_n)H_{m-1}(x_1,x_2) + H_2(x_3,\dots,x_n)H_{m-2}(x_1,x_2,x_3) \\ &\quad + \cdots + x_n^{n-1}H_{m-n+1}(x_1,\dots,x_n) \end{aligned} \tag{3.4}$$

and

$$H_m(x_1,\dots,x_{n-1},x_n) - H_m(x_1,\dots,x_{n-1},x_{n+1}) = (x_n - x_{n+1})H_{m-1}(x_1,\dots,x_n,x_{n+1}). \tag{3.5}$$

Equipped with the theoretical framework concerning symmetric functions, we will now show how we can make use of them investigating the symmetric cofactor of the difference product in an alternant.

3.1.3 Expressing simple Alternants as complete symmetric functions

At first we will consider alternants in which the generating functions are (nonnegative integer) powers of the variables. Hence, we are interested in alternants of the form

$$|A(x_1^{p_1}, x_2^{p_2}, \dots, x_n^{p_n})| = \begin{vmatrix} x_1^{p_1} & x_1^{p_2} & \cdots & x_1^{p_n} \\ x_2^{p_1} & x_2^{p_2} & \cdots & x_2^{p_n} \\ \vdots & \vdots & & \vdots \\ x_n^{p_1} & x_n^{p_2} & \cdots & x_n^{p_n} \end{vmatrix}$$

This type of alternant is known as *simple* alternant. (Note that negative integer powers can be reduced to nonnegative powers by taking out an appropriate factor).

We will now present an early result of Jacobi which expresses a simple alternant in terms of complete symmetric functions.

Theorem 14 *The quotient of any simple alternant by the corresponding difference product is expressible as a determinant whose elements are complete symmetric functions of the variables:*

$$\frac{|A(x_1^{p_1},\dots,x_n^{p_n})|}{DP(x_1,\dots,x_n)} = \begin{vmatrix} H_{p_1}(x_1,\dots,x_n) & H_{p_2}(x_1,\dots,x_n) & \cdots & H_{p_n}(x_1,\dots,x_n) \\ H_{p_1-1}(x_1,\dots,x_n) & H_{p_2-1}(x_1,\dots,x_n) & \cdots & H_{p_n-1}(x_1,\dots,x_n) \\ \vdots & \vdots & & \vdots \\ H_{p_1-n+1}(x_1,\dots,x_n) & H_{p_2-n+1}(x_1,\dots,x_n) & \cdots & H_{p_n-n+1}(x_1,\dots,x_n) \end{vmatrix} \tag{3.6}$$

Proof.

Subtracting the first row of $|A(x_1^{p_1},\dots,x_n^{p_n})|$ from all the following rows, we get:

$$|A(x_1^{p_1},\dots,x_n^{p_n})| = \begin{vmatrix} x_1^{p_1} & x_1^{p_2} & \cdots & x_1^{p_n} \\ x_2^{p_1} - x_1^{p_1} & x_2^{p_2} - x_1^{p_2} & \cdots & x_2^{p_n} - x_1^{p_n} \\ \vdots & \vdots & & \vdots \\ x_n^{p_1} - x_1^{p_1} & x_n^{p_2} - x_1^{p_2} & \cdots & x_n^{p_n} - x_1^{p_n} \end{vmatrix}.$$

Since $x_i^{p_j} - x_1^{p_j} = H_{p_j}(x_i) - H_{p_j}(x_1)$ for $i = 2,\dots,n$ we see that $x_i^{p_j} - x_1^{p_j} = (x_i - x_1)H_{p_j-1}(x_1,x_i)$ using identity (3.4). Thus, the factors $x_2 - x_1, x_3 - x_1, \dots, x_n - x_1$ may be taken out and we receive

$$\frac{|A(x_1^{p_1},\dots,x_n^{p_n})|}{\prod_{1<j\le n}(x_j - x_1)} = \begin{vmatrix} x_1^{p_1} & x_1^{p_2} & \cdots & x_1^{p_n} \\ H_{p_1-1}(x_1,x_2) & H_{p_2-1}(x_1,x_2) & \cdots & H_{p_n-1}(x_1,x_2) \\ \vdots & \vdots & & \vdots \\ H_{p_1-1}(x_1,x_n) & H_{p_2-1}(x_1,x_n) & \cdots & H_{p_n-1}(x_1,x_n) \end{vmatrix}.$$

Treating the resulting determinant in the same way, the elements of the second row being now the subtrahends, we apply identity (3.4) again, take out the factors $x_3 - x_2, x_4 - x_2, \dots, x_n - x_2$ and continue this process on the resulting determinant.

Finally we get

$$\frac{|A(x_1^{p_1}, \dots, x_n^{p_n})|}{\prod_{1 \le i < j \le n}(x_j - x_i)} \begin{vmatrix} x_1^{p_1} & x_1^{p_2} & \cdots & x_1^{p_n} \\ H_{p_1-1}(x_1, x_2) & H_{p_2-1}(x_1, x_2) & \cdots & H_{p_n-1}(x_1, x_2) \\ H_{p_1-2}(x_1, x_2, x_3) & H_{p_2-2}(x_1, x_2, x_3) & \cdots & H_{p_n-2}(x_1, x_2, x_3) \\ \vdots & \vdots & & \vdots \\ H_{p_1-n+1}(x_1, \dots, x_n) & H_{p_2-n+1}(x_1, \dots, x_n) & \cdots & H_{p_n-n+1}(x_1, \dots, x_n) \end{vmatrix}.$$

Multiplying the determinant on the right–hand side by unity in the form

$$\begin{vmatrix} 1 & 0 & 0 & \cdots & 0 \\ H_1(x_2, \dots, x_n) & 1 & 0 & & 0 \\ H_2(x_3, \dots, x_n) & H_1(x_3, \dots, x_n) & 1 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 0 \\ x_n^{n-1} & x_n^{n-2} & x_n^{n-3} & \cdots & 1 \end{vmatrix}$$

from the right and using identity (3.5), we obtain the required result.

$\square$

How can we benefit from this result?

We will show that we can derive arbitrary order determinant formulas for a certain class of simple alternants. Let us illustrate the theorem in an example.

Example

Consider the simple alternant $|A(f_1, \dots, f_n)|$ of order n generated by the functions $f_j = x_i^{j-1}$ for $j = 1, \dots, n-1$ and $f_n = x_i^s$. The theorem tells us (writing H_m instead of $H_m(x_1, \dots, x_n)$ for short) that

$$\frac{|A(f_1, \dots, f_n)|}{\mathrm{DP}(x_1, \dots, x_n)} = \begin{vmatrix} 1 & H_1 & H_2 & H_{n-2} & H_s \\ 0 & 1 & H_1 & \vdots & \vdots \\ 0 & 0 & \ddots & H_1 & \\ \vdots & \vdots & \ddots & 1 & H_{s-n+2} \\ 0 & 0 & \cdots & 0 & H_{s-n+1} \end{vmatrix} = H_{s-n+1}.$$

Now we want to generalize this for simple alternants generated by functions of the form $f_j = x_i^{j+d}$ for $j = 1, \dots, n-k$ and $d, k \in \mathbb{N}$, and arbitrary monomial functions f_j for $j = n-k+1, \dots, n$.

We simply factor out $\prod_{i=1}^{n} x_i^{d+1}$, apply Theorem 14 and see by expansion of the first $n-k$ columns that the order n cofactor of the difference product reduces to a determinant of fixed integer order k with complete symmetric function entries.

However, it is not possible to derive an explicit formula for alternants like $A(x_1^0, x_2^2, x_3^3, \dots, x_n^n)$ using this method which is very unsatisfactory. We will see later, how formulas for these alternants can be derived using elementary symmetric functions.

Mitchell [Mit81] shows how it is possible to derive a summation formula that describes the expanded determinant without the use of complete symmetric functions:

We slightly change the setting: We have simple alternants of the form $|A(x_1^{p_1}, \ldots, x_n^{p_n})|$ with $p_1 \geq \cdots \geq p_n \geq 0$. A trivial adaption of Theorem 14, gives us

$$\frac{|A(x_1^{p_1}, x_2^{p_2}, \ldots, x_n^{p_n})|}{\mathrm{DP}(x_1, \ldots, x_n)} = \prod_{i=1}^{n} x_i^{p_n} \begin{vmatrix} H_{p_1-p_n-n+1} & H_{p_2-p_n-n+1} & \cdots & H_{p_{n-1}-p_n-n+1} \\ \vdots & \vdots & & \vdots \\ H_{p_1-p_n-2} & H_{p_2-p_n-2} & \cdots & H_{p_{n-1}-p_n-2} \\ H_{p_1-p_n-1} & H_{p_2-p_n-1} & \cdots & H_{p_{n-1}-p_n-1} \end{vmatrix}_n$$

where the H's involve $x_1, \ldots, x_n$.

Expanding each element of the right–hand determinant by means of (3.3) and resolving the determinant into the sum of similar determinants using multilinearity, we get

$$\frac{|A(x_1^{p_1}, x_2^{p_2}, \ldots, x_n^{p_n})|}{\mathrm{DP}(x_1, \ldots, x_n)} = \prod_{i=1}^{n} x_i^{p_n} \sum_{l_1=0}^{p_1-p_n-1} \sum_{l_2=0}^{p_2-p_n-1} \cdots \sum_{l_{n-1}=0}^{p_{n-1}-p_n-1} x_n^{\alpha} \begin{vmatrix} H_{l_1-n+2} & H_{l_2-n+2} & \cdots & H_{l_{n-1}-n+2} \\ \vdots & \vdots & & \vdots \\ H_{l_1-1} & H_{l_2-1} & \cdots & H_{l_{n-1}-1} \\ H_{l_1} & H_{l_2} & \cdots & H_{l_{n-1}} \end{vmatrix}_{n-1}$$

where the H's now involve only $x_1, \ldots, x_{n-1}$ and $\alpha = \sum_{i=1}^{n} p_i - n - 1 - np_1 - \sum_{i=1}^{n-1} l_i$.

A little thought shows that the lower limits of l_i may be put to $p_{i+1} - p_n$ for $i = 1, \ldots, n-2$ (see [Mit81] for details).

Hence, we are left with

$$\frac{|A(x_1^{p_1}, x_2^{p_2}, \ldots, x_n^{p_n})|}{\mathrm{DP}(x_1, \ldots, x_n)} = \prod_{i=1}^{n} x_i^{p_n} \sum_{l_1=p_2-p_n}^{p_1-p_n-1} \sum_{l_2=p_3-p_n}^{p_2-p_n-1} \cdots \sum_{l_{n-2}=p_{n-1}-p_n}^{p_{n-2}-p_n-1} \sum_{l_{n-1}=0}^{p_{n-1}-p_n-1} x_n^{\alpha} \frac{|A(x_1^{l_1}, \ldots, x_{n-1}^{l_{n-1}})|}{\mathrm{DP}(x_1, \ldots, x_{n-1})}$$

and have a recursive summation formula.

Mitchell [Mit81] also discusses the number of monomials in the cofactor of a simple alternant, that is the number of monomials in the symmetric function:

Lemma 5 *The number of terms in*

$$\frac{\begin{vmatrix} x_1^{p_1} & x_1^{p_2} & \cdots & x_1^{p_n} \\ x_2^{p_1} & x_2^{p_2} & \cdots & x_2^{p_n} \\ \vdots & \vdots & & \vdots \\ x_n^{p_1} & x_n^{p_2} & \cdots & x_n^{p_n} \end{vmatrix}}{DP(x_1, \ldots, x_n)},$$

where w.l.o.g. $p_1 < p_2 < \cdots < p_n$, is

$$\frac{\begin{vmatrix} 1 & p_1 & \cdots & p_1^{n-1} \\ 1 & p_2 & \cdots & p_2^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & p_n & \cdots & p_n^{n-1} \end{vmatrix}}{\begin{vmatrix} 0^0 & 0^1 & \cdots & 0^{n-1} \\ 1^0 & 1^1 & \cdots & 1^{n-1} \\ \vdots & \vdots & & \vdots \\ (n-1)^0 & (n-1)^1 & \cdots & (n-1)^{n-1} \end{vmatrix}} = \frac{\prod_{1 \leq i < j \leq n} (p_j - p_i)}{\prod_{i=1}^{n-1} i!}.$$

Proof.

Since the number of terms in $H_m(x_1, \ldots, x_n)$ is $\binom{m+n-1}{n-1}$, Theorem 14 tells us that there are

$$\begin{vmatrix} \binom{p_1+n-1}{n-1} & \binom{p_2+n-1}{n-1} & \cdots & \binom{p_n+n-1}{n-1} \\ \binom{p_1+n-2}{n-1} & \binom{p_2+n-2}{n-1} & \cdots & \binom{p_n+n-2}{n-1} \\ \vdots & \vdots & & \vdots \\ \binom{p_1}{n-1} & \binom{p_2}{n-1} & \cdots & \binom{p_n}{n-1} \end{vmatrix}$$

possible terms for the cofactor which can be seen setting the x_i to 1. Using the identity $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$ on the first $n-1$ rows and subtracting consecutive rows we get

$$\begin{vmatrix} \binom{p_1+n-2}{n-2} & \binom{p_2+n-2}{n-2} & \cdots & \binom{p_n+n-2}{n-2} \\ \binom{p_1+n-3}{n-2} & \binom{p_2+n-3}{n-2} & \cdots & \binom{p_n+n-3}{n-2} \\ \vdots & \vdots & & \vdots \\ \binom{p_1}{n-2} & \binom{p_2}{n-2} & \cdots & \binom{p_n}{n-2} \\ \binom{p_1}{n-1} & \binom{p_2}{n-1} & \cdots & \binom{p_n}{n-1} \end{vmatrix}.$$

Repeating this, we eventually end up with

$$\begin{vmatrix} \binom{p_1}{0} & \binom{p_2}{0} & \cdots & \binom{p_n}{0} \\ \binom{p_1}{1} & \binom{p_2}{1} & \cdots & \binom{p_n}{1} \\ \vdots & \vdots & & \vdots \\ \binom{p_1}{n-2} & \binom{p_2}{n-2} & \cdots & \binom{p_n}{n-2} \\ \binom{p_1}{n-1} & \binom{p_2}{n-1} & \cdots & \binom{p_n}{n-1} \end{vmatrix}.$$

This resembles a Vandermonde type determinant. Let us expand the binomial coefficients:

$$\binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k!} = \frac{1}{k!}n^k + P(n^{k-1}),$$

where $P(n^{k-1})$ is a polynomial in n with leading power $k-1$. Hence we have

$$\begin{vmatrix} 1 & 1 & \cdots & 1 \\ p_1 & p_2 & \cdots & p_n \\ \frac{1}{2!}p_1^2 + P(p_1) & \frac{1}{2}p_2^2 + P(p_2) & \cdots & \frac{1}{2}p_n^2 + P(p_n) \\ \vdots & \vdots & & \vdots \\ \frac{1}{(n-1)!}p_1^{n-1} + P(p_1^{n-1}) & \frac{1}{(n-1)!}p_2^{n-1} + P(p_2^{n-2}) & \cdots & \frac{1}{(n-1)!}p_n^{n-1} + P(p_n^{n-2}) \end{vmatrix}.$$

Factoring out $(n-i)!$ in each row i and observing that the $P(\cdot)$ summand does not contribute to the determinant, we finally get

$$\frac{\begin{vmatrix} 1 & p_1 & \cdots & p_1^{n-1} \\ 1 & p_2 & \cdots & p_2^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & p_n & \cdots & p_n^{n-1} \end{vmatrix}}{\prod_{i=1}^{n-1}(n-i)!} = \frac{\prod_{1 \le i < j \le n}(p_j - p_i)}{\prod_{i=1}^{n-1} i!}.$$

$\square$

3.1.4 Reducing alternants to simple alternants

In the previous subsection, we restricted ourselves to simple alternants with monomial entries and derived formulas for special cases of symbolic order. Now we will show how we can express general alternants with polynomial entries as a combination of simple alternants.

Assume an order n alternant A with generating functions $f_j(x_i) = a_{0j} + a_{1j}x_i + a_{2j}x_i^2 + \cdots + a_{rj}x_i^r$.

Then we can conclude from the multilinearity of determinants, that

$$|A(f_1(x_1), \ldots, f_n(x_n))| = \sum_{l_1=1}^{r} \sum_{l_2=1}^{r} \cdots \sum_{l_n=1}^{r} a_{l_1,1} a_{l_2,2} \cdots a_{l_n,n} \begin{vmatrix} x_1^{l_1} & x_1^{l_2} & \cdots & x_1^{l_n} \\ x_2^{l_1} & x_2^{l_2} & \cdots & x_2^{l_n} \\ \vdots & \vdots & & \vdots \\ x_n^{l_1} & x_n^{l_2} & \cdots & x_n^{l_n} \end{vmatrix}.$$

Example

Consider $|A|$ with $f_1(x_i) = a + bx_i,\ f_2(x_i) = x_i,\ f_3(x_i) = cx_i^2 - dx_i^3$.

$$|A| \begin{vmatrix} a + bx_1 & x_1 & cx_1^2 - dx_1^3 \\ a + bx_2 & x_2 & cx_2^2 - dx_2^3 \\ a + bx_3 & x_3 & cx_3^2 - dx_3^3 \end{vmatrix}.$$

We get

$$|A| = ac \begin{vmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{vmatrix} + bc \begin{vmatrix} x_1 & x_1 & x_1^2 \\ x_2 & x_2 & x_2^2 \\ x_3 & x_3 & x_3^2 \end{vmatrix} - ad \begin{vmatrix} 1 & x_1 & x_1^3 \\ 1 & x_2 & x_2^3 \\ 1 & x_3 & x_3^3 \end{vmatrix} - bd \begin{vmatrix} x_1 & x_1 & x_1^3 \\ x_2 & x_2 & x_2^3 \\ x_3 & x_3 & x_3^3 \end{vmatrix}$$

We observe that the second and the last simple alternant vanish, thus, using Theorem 14, we obtain the result

$$|A| = (ac - adH_1(x_1, x_2, x_3))\mathrm{DP}(x_1, \ldots, x_n).$$

Obviously, a reduction to a fixed integer number of simple alternants is only possible if all but a fixed integer number of generating functions of the alternant are monomials. Unfortunately, we need $m_1 m_2 \cdots m_n$ simple alternants which is exponential in n. However, we have already seen in the example, that many vanishing simple alternants (since columns coincide) are included in the summation. It is suggested to do the summation in a reverse order, first considering the bigger powers. If then the biggest power in a combination $p_{l_1}^{(1)}, \ldots, p_{l_n}^{(n)}$ is less than $n - 1$, resulting in a zero simple alternant since there have to be two identical columns, we can abort the summation since the maximum degree of later combinations can only be smaller.

3.1.5 Computing elementary symmetric function representation of alternants

After we have seen how the cofactor of simple alternants and the difference product can be expressed by a determinant of the same order involving complete symmetric functions we would like a similar result for the elementary symmetric functions.

We will even show a result from [Met60], that the cofactor of any alternant with polynomial entries and its difference product may be expressed as a determinant involving coefficients of the alternant elements and elementary symmetric functions. However, the order will be the highest degree of the generating polynomials.

Theorem 15 *Let $|A|$ be an alternant of order n with column generating functions*

$$f_j(x_i) = a_{oj} + a_{1j}x_i + a_{2j}x_i^2 + \cdots + a_{rj}x_i^r, \ \ (r \geq n)$$

and let $S_k = (-1)^k \sigma_k$ for $k = 0, \ldots, n$ (where σ_k is the abbreviation of $\sigma_k(x_1, \ldots, x_n)$ and $\sigma_0 := 1$). The following identity holds:

$$\frac{|A(f_1(x_1), \ldots, f_n(x_n))|}{DP(x_1, \ldots, x_n)} = \begin{vmatrix} a_{01} & a_{11} & a_{21} & \cdots & a_{n1} & a_{n+1,1} & \cdots & a_{r1} \\ a_{02} & a_{12} & a_{22} & \cdots & a_{n2} & a_{n+1,2} & \cdots & a_{r2} \\ \vdots & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ a_{0n} & a_{1n} & a_{2n} & \cdots & a_{nn} & a_{n+1,n} & \cdots & a_{rm} \\ S_n & S_{n-1} & \cdots & S_1 & S_0 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & S_n & S_{n-1} & \cdots & S_1 & S_0 & 0 \\ 0 & \cdots & 0 & S_n & S_{n-1} & \cdots & S_1 & S_0 \end{vmatrix}_{r+1} . \tag{3.7}$$

Proof.

We denote the right–hand determinant by Δ. Starting with Δ, we try to get to the left–hand side of the identity.

$$\Delta \cdot \mathrm{DP}(x_1, \ldots, x_n, \omega_0, \ldots, \omega_{r-n}) = \Delta \cdot \begin{vmatrix} 1 & \cdots & 1 & 1 & \cdots & 1 \\ x_1 & \cdots & x_n & \omega_0 & \cdots & \omega_{r-n} \\ \vdots & & \vdots & \vdots & & \vdots \\ x_1^n & \cdots & x_n^n & \omega_0^n & \cdots & \omega_{r-n}^n \\ \vdots & & \vdots & \vdots & & \vdots \\ x_1^r & \cdots & x_n^r & \omega_0^r & \cdots & \omega_{r-n}^r \end{vmatrix} .$$

Multiplying yields

$$= \begin{vmatrix} f_1(x_1) & f_1(x_2) & \cdots & f_1(x_n) & f_1(\omega_0) & \cdots & f_1(\omega_{r-n}) \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ f_n(x_1) & f_n(x_2) & \cdots & f_n(x_n) & f_n(\omega_0) & \cdots & f_n(\omega_{r-n}) \\ \phi(x_1) & \phi(x_2) & \cdots & \phi(x_n) & \phi(\omega_0) & \cdots & \phi(\omega_{r-n}) \\ x_1\phi(x_1) & x_2\phi(x_2) & \cdots & x_n\phi(x_n) & \omega_0\phi(\omega_0) & \cdots & \omega_{r-n}\phi(\omega_{r-n}) \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ x_1^{r-n}\phi(x_1) & x_2^{r-n}\phi(x_2) & \cdots & x_n^{r-n}\phi(x_n) & \omega_0^{r-n}\phi(\omega_0) & \cdots & \omega_{r-n}^{r-n}\phi(\omega_{r-n}) \end{vmatrix}_{r+1} ,$$

where for shortness we put

$$\phi(x) = S_0 x^n + S_1 x^{n-1} + \cdots S_{n-1}x + S_n = (x - x_1) \cdots (x - x_n).$$

Since $\phi(x_i) = 0$ for all $i = 1, \ldots, n$, this determinant is a triangular block determinant and can be retransformed into a determinant product:

$$= \begin{vmatrix} f_1(x_1) & f_1(x_2) & \cdots & f_1(x_n) \\ f_2(x_1) & f_2(x_2) & \cdots & f_2(x_n) \\ \vdots & \vdots & & \vdots \\ f_n(x_1) & f_n(x_2) & \cdots & f_n(x_n) \end{vmatrix} \cdot \begin{vmatrix} \phi(\omega_0) & \phi(\omega_1) & \cdots & \phi(\omega_{r-n}) \\ \omega_0\phi(\omega_0) & \omega_1\phi(\omega_1) & \cdots & \omega_{r-n}\phi(\omega_{r-n}) \\ \vdots & \vdots & & \vdots \\ \omega_0^{r-n}\phi(\omega_0) & \omega_1^{r-n}\phi(\omega_1) & \cdots & \omega_{r-n}^{r-n}\phi(\omega_{r-n}) \end{vmatrix} .$$

It is obvious that the first determinant in this product is the transposed alternant $|A(f_1(x_1), \dots, f_n(x_n))|$. Factoring out $\phi(w_i),\ i = 0, \dots, n$ we see that the resulting second determinant is the difference product of $\omega_0, \dots, \omega_{r-n}$. Hence, we get

$$= |A(f_1(x_1), \dots, f_n(x_n))|\, \phi(\omega_0) \cdots \phi(\omega_{r-n})\, DP(\omega_0, \dots, \omega_{r-n}).$$

Since we can write $\mathrm{DP}(x_1, \dots, x_n, \omega_0, \dots, \omega_{r-n})$, the difference product which we multiplied to Δ at the start of the proof, in the form $\mathrm{DP}(x_1, \dots, x_n)\mathrm{DP}(\omega_0, \dots, \omega_{r-n})\phi(\omega_0) \cdots \phi(\omega_{r-n})$, we can remove common factors and are left with

$$\Delta \cdot \mathrm{DP}(x_1, \dots, x_n) = |A(f_1(x_1), \dots, f_n(x_n))|,$$

as was to be proved.

$\square$

Note that the order $r+1$ of the cofactor determinant depends upon the maximum degree of the generating functions of the alternant and is not simply of order n like in the case of complete symmetric functions. We also observe that the alternant vanishes for $r < n-1$ and that the cofactor determinant contains only the coefficients for $r = n-1$.

Let us consider the special case of simple alternants again. Let $|A|$ be a simple alternant with generating functions $f_1(x_i) = x_i^{p_1}, \dots, f_n(x_i) = x_i^{p_n}$. Assume w.l.o.g. that $p_1 \le \cdots \le p_n = r$. The foregoing theorem then yields the following structure of the cofactor determinant:

$$\frac{|A|}{DP(x_1, \dots, x_n)} = \left|\begin{array}{cccccccccc} 0 & \cdots & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \vdots & & \vdots & \vdots & & \vdots & 0 & \ddots & & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & & 0 & 1 \\ S_n & & & & \cdots & & & S_0 & & \mathbf{0} \\ & \ddots & & & & & & & \ddots & \\ \mathbf{0} & & S_n & & & & \cdots & & & S_0 \end{array}\right|_{r+1} =: \Delta,$$

where the 1 entries resulting from generating function f_j are in column $p_j + 1$.

Since the first n rows have only one nonzero element, the entry 1 at position $p_j + 1$, we can reduce Δ to a determinant Δ' of order $r - n + 1$ by expanding the first n rows (recall the minor notation of the first chapter):

$$\Delta' = (-1)^{p_1 + \cdots + p_n - \frac{n(n-1)}{2}} \Delta_{(1,2,\dots,n),(p_1+1, p_2+1, \dots, p_n+1)}.$$

Obviously, the columns thrown out of Δ are the $p_1 + 1$st, the $p_2 + 1$st … and the $p_n + 1$st (i.e. the last column). How do we get the signfactor? Expanding the first row, we get $(-1)^{p_1}$, the second row $(-1)^{p_2 - 1}$, the next $(-1)^{p_3 - 2}$, etc. which establishes our result.

Example

Consider the simple alternant $|A|$ given by column generating functions $f_j(x_i) = x_i^{j-1}$ for $j = 1, \dots, k-1$ and $f_j(x_i) = x_i^j$ for $j = k, \dots, n$.

Then we have

$$\frac{|A|}{\mathrm{DP}(x_1,\dots,x_n)} = \begin{vmatrix} 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \ddots & 1 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 1 & \ddots & \vdots \\ \vdots & & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 & 1 \\ S_n & \cdots & S_{n-k+1} & S_{n-k} & S_{n-k-1} & \cdots & S_0 \end{vmatrix} = (-1)^{n-k} S_{n-k} = \sigma_{n-k}.$$

We observe that we are able to express symbolic order alternants in terms of a formula containing combinations of elementary symmetric functions that we could not express in terms of complete symmetric functions (as the example shows for $k \in \mathbb{N}$).

In fact, we can compute an explicit formula for the cofactor of all simple alternants with generating functions of maximum degree $n+d,\ d \in \mathbb{N}$, since for these degrees it is possible to reduce the order of the cofactor determinant in Theorem 15 to a fixed integer order via expansion of the first n rows.

Computing formulas of easy polynomial alternants We already showed that we can handle simple alternants with maximum degree $n+d,\ d \in \mathbb{N}$. In subsection 3.1.4, we established that polynomial alternants can be reduced to a combination of a fixed number of simple alternants if only a fixed number of generating functions are polynomials. We will now discuss, how it is possible to obtain formulas containing elementary symmetric functions for certain polynomial alternants of relatively easy structure:

First we look at alternants with generating functions $p_j(x_i) = c_1 x_i^{l_1} + \cdots + c_k x_i^{l_k} + c_{diag} x_i^{j+d}$ (see the example below) with

$k, d \in \mathbb{N}$, the coefficients $c_t \neq 0$ and the powers l_t being constants (i.e. independent of j) with $-m \leq l_t \leq n+m$, m integer. Applying Theorem 15 we see that the cofactor of the difference product is of the form

$$\begin{vmatrix} 0 & c_1 & 0 & c_{diag} & 0 & c_2 & 0 & \cdots & 0 & c_k \\ 0 & c_1 & 0 & 0 & c_{diag} & c_2 & 0 & \cdots & 0 & c_k \\ 0 & c_1 & 0 & 0 & 0 & c_{diag}+c_2 & 0 & \cdots & 0 & c_k \\ 0 & c_1 & 0 & 0 & 0 & c_2 & c_{diag} & \ddots & \vdots & c_k \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \mathbf{0} & \ddots & 0 & \vdots \\ 0 & c_1 & 0 & 0 & 0 & c_2 & & & c_{diag} & c_k \\ S_n & S_{n-1} & & \cdots & & & S_0 & 0 & \cdots & 0 \\ 0 & S_n & S_{n-1} & & \cdots & & & S_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & & & & & & \ddots & 0 \\ 0 & \cdots & 0 & S_n & S_{n-1} & & \cdots & & & S_0 \end{vmatrix}.$$

Observe that we have a diagonal with vertical lines in the coefficient block. Our aim is to eliminate these vertical lines corresponding to the constant monomial summands:

1. Subtract a suitable multiple of the first column corresponding to the minimal constant power from the other columns corresponding to constant powers in order to reduce them to zero.
2. Subtract the remaining diagonal columnwise from the remaining vertical line, such that only the diagonal remains.
3. Expand the resulting determinant rowwise. Compute the resulting determinant of fixed integer order.

There are some subtleties to be considered: If the first constant power column is located within the diagonal then we have the sum $c_{diag} + c_1$ at the intersection element located in row w, say. Thus every subtraction eliminates the next desired column part except element w of the column. We eliminate these rests adding appropriate multiples of corresponding rows. Observe that we do not really need to perform step 2. if the smallest constant power lies within the diagonal, since expansion would eliminate the altered S_k.

Example

Consider $|A| = |(1 + 2x_i^4 - 3x_i^{n-1} + x_i^{j+1})|_n$. Theorem 15 gives the cofactor

$$\begin{vmatrix}
1 & 0 & 1 & 0 & 2 & 0 & \cdots & 0 & -3 & 0 & 0 \\
1 & 0 & 0 & 1 & 2 & 0 & \cdots & 0 & -3 & 0 & 0 \\
1 & 0 & 0 & 0 & 3 & 0 & \cdots & 0 & -3 & 0 & 0 \\
1 & 0 & 0 & 0 & 2 & 1 & \ddots & \vdots & -3 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & 0 & \ddots & 0 & \vdots & \vdots & \vdots \\
1 & 0 & 0 & 0 & 2 & \vdots & \ddots & 1 & -3 & 0 & 0 \\
1 & 0 & 0 & 0 & 2 & 0 & \cdots & 0 & -2 & 0 & 0 \\
1 & 0 & 0 & 0 & 2 & 0 & \cdots & 0 & -3 & 1 & 0 \\
1 & 0 & 0 & 0 & 2 & 0 & \cdots & 0 & -3 & 0 & 1 \\
S_n & S_{n-1} & S_{n-2} & S_{n-3} & S_{n-4} & S_{n-5} & \cdots & S_2 & S_1 & S_0 & 0 \\
0 & S_n & S_{n-1} & S_{n-2} & S_{n-3} & S_{n-4} & \cdots & S_3 & S_2 & S_1 & S_0
\end{vmatrix}.$$

Performing step 1. yields

$$\begin{vmatrix}
1 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots \\
1 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1 \\
S_n & S_{n-1} & S_{n-2} & S_{n-3} & S_{n-4} - 2S_n & S_{n-5} & \cdots & S_2 & S_1 + 3S_n & S_0 & 0 \\
0 & S_n & S_{n-1} & S_{n-2} & S_{n-3} & S_{n-4} & \cdots & S_3 & S_2 & S_1 & S_0
\end{vmatrix},$$

Step 2. gives

$$\begin{vmatrix}
0 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1 \\
-\sum_{k=0}^{n-2} S_k & S_{n-1} & S_{n-2} & S_{n-3} & S_{n-4} - 2S_n & S_{n-5} & \cdots & S_2 & S_1 + 3S_n & S_0 & 0 \\
-\sum_{k=0}^{n-1} S_k & S_n & S_{n-1} & S_{n-2} & S_{n-3} & S_{n-4} & \cdots & S_3 & S_2 & S_1 & S_0
\end{vmatrix}.$$

After the rowwise expansion in step 3. we are left with the cofactor

$$\begin{vmatrix}
-\sum_{k=0}^{n-2} S_k & S_{n-1} \\
\sum_{k=0}^{n-1} S_k & S_n
\end{vmatrix} = -(S_n \sum_{k=0}^{n-2} S_k \, + \, S_{n-1} \sum_{k=0}^{n-1} S_k).$$

Next, we focus on alternants of the form $|A| = |(\sum_{l=0}^{k} c_l x_i^{j+p_l})|_n$, with $k, p_l \in \mathbb{N}$ and constant $c_l \neq 0$. Theorem 15 gives us the cofactor of the difference product which now has the following form:

$$\begin{vmatrix} c_0 & 0 & \cdots & 0 & c_1 & 0 & \cdots & \cdots & 0 & c_k & 0 & \cdots & 0 \\ 0 & c_0 & 0 & \cdots & 0 & c_1 & 0 & \cdots & \cdots & 0 & c_k & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & c_0 & 0 & \cdots & 0 & c_1 & 0 & \cdots & \cdots & 0 & c_k \\ S_n & & & & & \cdots & & & & & S_0 & & \mathbf{0} \\ & \ddots & & & & & & & & & & \ddots & \\ \mathbf{0} & & S_n & & & \cdots & & & & & & & S_0 \end{vmatrix}.$$

The highest power determines the order of the cofactor determinant. We will reduce its coefficient block to a diagonal by successively eliminating the other diagonals bottom up, i.e. assuming $p_1 < p_2 < \cdots < p_k = r$ we perform the column operations $col(p_1+1+i) = col(p_1+1+i) - \frac{c_1}{c_k} col(p_k+1+i), \ldots, col(p_{k-1}+1+i) = col(p_{k-1}+1+i) - \frac{c_{k-1}}{c_k} col(p_k+1+i)$ for $i = n, n-1, \ldots, 2, 1$.

After this reduction, we can expand the cofactor determinant row–wise and get a determinant of fixed integer order. But what about its entries ?

Unfortunately, the entries obey nasty recurrences and we are not able to find a closed form for these recurrences in general. Restricting $k = 2$ (that is allowing only two monomials with exponents containing j), however, we can specify the entries explicitly and hence find useful formulas for the cofactor determinant:

Assume, we have $|A| = |(c_1 x_i^{j-1} + c_2 x_i^{j-1+d})|_n$, $d \geq 1$.

$$|A| = \begin{vmatrix} c_1 & 0 & \cdots & 0 & c_2 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & c_1 & 0 & \cdots & 0 & c_2 \\ S_n & \cdots & \cdots & \cdots & S_0 & 0 & \cdots & 0 \\ 0 & \ddots & & & & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & & & \ddots & 0 \\ 0 & \cdots & 0 & S_n & \cdots & \cdots & \cdots & S_0 \end{vmatrix}.$$

We define $S_{-k} = 0$ and $S_{n+k} = 0$ for $k > 1$. Eliminating the left diagonal, resulting from $c_1 x_i^{j-1}$, in the cofactor determinant has the following effect on the S_l band: The elements of the last row from right to left are now generated by

$$\sum_{l=0}^{\lfloor \frac{k}{d} \rfloor} (-\frac{c_1}{c_2})^l S_{k-dl}$$

for $k = 0, \ldots, r$. The elements of the penultimate row form right to left are generated by $\sum_{l=0}^{\lfloor \frac{k}{d} \rfloor} (-\frac{c_1}{c_2})^l S_{k-dl-1}$ and so on.

Thus, we obtain a fixed integer order determinant after expanding the first n rows and may compute an explicit formula for the cofactor.

Example

Consider $|A| = |(x_i^{j-1} + 2x_i^{j+1})|_n$. The cofactor of the difference product is

$$\begin{vmatrix} 1 & 0 & 2 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & 0 & 2 \\ S_n & & \cdots & & S_0 & 0 \\ 0 & S_n & & \cdots & & S_0 \end{vmatrix}.$$

Eliminating the left diagonal, we obtain

$$\begin{vmatrix} 0 & 0 & 2 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 2 & \ddots & \vdots \\ \vdots & \vdots & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 2 \\ \sum_{l=0}^{\lfloor \frac{n}{2} \rfloor} (-\frac{1}{2})^l S_{n-2l} & \sum_{l=1}^{\lfloor \frac{n}{2} \rfloor} (-\frac{1}{2})^{l-1} S_{n-2l+1} & \cdots & S_1 & S_0 & 0 \\ \sum_{l=1}^{\lfloor \frac{n}{2} \rfloor} (-\frac{1}{2})^l S_{n-2l+1} & \sum_{l=0}^{\lfloor \frac{n}{2} \rfloor} (-\frac{1}{2})^l S_{n-2l} & \cdots & S_2 - \frac{1}{2}S_0 & S_1 & S_0 \end{vmatrix}.$$

Expanding yields the cofactor formula

$$2^n((\sum_{l=0}^{\lfloor \frac{n}{2} \rfloor}(-\frac{1}{2})^l S_{n-2l})^2 + (\sum_{l=1}^{\lfloor \frac{n}{2} \rfloor}(-\frac{1}{2})^l S_{n-2l+1})^2).$$

The combination of the two approaches would be too complicated to describe since several special cases would have to be considered.

3.1.6 Taking care of fractions and transcendental or exponential generating functions

Recall that we imposed a restriction on the alternant elements (hence on the generating functions), they had to be polynomials.

What can we do, if this property is not given ?

In the case of fractions, we simply clear the fractions extracting an appropriate factor and attempt to solve the resulting alternant using one of the given theorems.

Transcendental generating functions (like $\sin, \cos$ etc.) or exponential functions impose more problems, however. If the generating functions are purely exponential or purely transcendental, it is suggested to substitute new variables and try to obtain polynomial elements. E.g. we substitute $u_i = \exp(x_i)$ or $u_i = \sin(x_i)$.

We will be making use of exponential and trigonometric identities, like

$$\exp(x + y) = \exp(x)\exp(y),$$

$$\sin^2(x) = 1 - \cos^2(x), \tag{3.8}$$

$$\sin(x \pm y) = \sin(x)\cos(y) \pm \cos(x)\sin(y), \tag{3.9}$$

$$\cos(x \pm y) = \cos(x)\sin(y) \mp \sin(x)\cos(y), \tag{3.10}$$

$$\cos(nx) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} (-1)^k \binom{n}{2k} \cos^{n-2k}(x)\sin^{2k}(x), \tag{3.11}$$

$$\sin(nx) = \sum_{k=0}^{\lfloor \frac{n-1}{2} \rfloor} (-1)^k \binom{n}{2k+1} \cos^{n-2k+1}(x)\sin^{2k+1}(x).$$

Many mixed or other complicated expressions cannot be transformed into polynomial form, so having automation in mind, we will just show how a simple trigonometric alternant can be transformed into a polynomial one, such that its formula can be obtained.

We will need the following lemma.

Lemma 6 *The following identity holds:*

$$\cos(nx) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} (-1)^k \cos^{n-2k}(x) \sum_{l=k}^{\lfloor \frac{n}{2} \rfloor} \binom{l}{k}\binom{n}{2l}.$$

Proof.

Plugging (3.8) in (3.11) we get

$$\begin{aligned}\cos(nx) = \cos^n(x) - \binom{n}{2}\cos^{n-2}(x)(1-\cos^2(x)) + \binom{n}{4}\cos^{n-4}(x)(1-\cos^2(x))^2 \\ - \binom{n}{6}\cos^{n-6}(x)(1-\cos^2(x))^3 + \cdots .\end{aligned} \tag{3.12}$$

Now we want to order the cosine powers. Let us enumerate the terms in

$$C^k := (1-\cos^2(x))^k = \sum_{l=0}^{k} (-1)^k \binom{k}{l} \cos^{2l}(x).$$

We observe that each term in (3.12) contributes to $\cos^n(x)$, namely, evidently the first term, the second term if we multiply with the second term in C^1, the third term if we multiply with the third term in C^2, and so on. Since the signs cancel out we get the following $\cos^n(x)$ cofactor

$$\binom{n}{0} + \binom{n}{2} + \binom{n}{4} + \cdots + \binom{n}{2\lfloor \frac{n}{2} \rfloor}.$$

Now we examine the cofactor of $\cos^{n-2}(x)$ and observe that from the second term on in (3.12), each one contributes to it, namely the second term if we multiply with the first term in C^1, the third term if we multiply with the second term in C^2, and so on. Hence we get the cofactor

$$-(\binom{n}{2}+2\binom{n}{4}+3\binom{n}{6}+\cdots+\left\lfloor\frac{n}{2}\right\rfloor\binom{n}{2\left\lfloor\frac{n}{2}\right\rfloor}).$$

Similarly, we get

$$\binom{2}{2}\binom{n}{4}+\binom{3}{2}\binom{n}{6}+\cdots$$

as cofactor of $\cos^{n-4}(x)$. Repeating the process yields the Lemma.

$\square$

Example

Consider the alternant $|A| = |(\cos((j-1)x_i))|_n$. We attempt to reduce this alternant to polynomial form, so we have to get rid of the $\cos((j-1)x_i)$ in some way and replace it with something like $\cos^{(j-1)}(x_i)$ which would allow us fruitful substitution.

Using Lemma 6, we get $\cos((j-1)x_i) = \cos^{(j-1)}(x_i)(1+\binom{j-1}{2}+\binom{j-1}{4}+\cdots+\binom{j-1}{m-2}+\binom{j-1}{m}) + P(\cos^{j-3}(x_i))$, where m is the biggest even number $\leq j-1$ and $P(\cos^{j-3}(x_i))$ is a polynomial in $\cos(x_i)$ with leading power $j-3$.

With the conclusions from the binomial identity, $\sum_{k=0}^{n}\binom{n}{k} = 2^n$ and $\sum_{k=0}^{n}(-1)^k\binom{n}{k} = 0$, we obtain

$$\binom{n}{0}+\binom{n}{2}+\cdots+\binom{n}{m} = 2^{n-1},$$

with m being the largest even number $\leq n$ (see [BS91]). Hence we may transform the given alternant into

$$|A| = \begin{vmatrix} 1 & \cos(x_1) & 2\cos^2(x_1)-1 & \cdots & 2^{n-2}\cos^{n-1}(x_1)+P(\cos^{n-3}(x_i)) \\ 1 & \cos(x_2) & 2\cos^2(x_2)-1 & \cdots & 2^{n-2}\cos^{n-1}(x_2)+P(\cos^{n-3}(x_i)) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \cos(x_n) & 2\cos^2(x_n)-1 & \cdots & 2^{n-2}\cos^{n-1}(x_n)+P(\cos^{n-3}(x_i)) \end{vmatrix}.$$

Substituting $u_i = \cos(x_i)$ we are left with a polynomial alternant in u_i. Factoring out $\prod_{i=1}^{n} 2^{i-2}$, we get the difference product of the variables u_i, since the terms of lower order do not contribute to the determinant formula (the cofactor determinant is a lower triangular determinant). Back substitution yields the determinant formula for the original alternant:

$$|A| = 2^{\frac{(n-1)(n-2)}{2}} \prod_{1\leq i<j\leq n}(\cos(x_j)-\cos(x_i)). \tag{3.13}$$

Analogous proceeding in the sine case establishes corresponding formulas.

It remains to note, that we cannot use the same trick for $|\cos(jx_i)|_n$ since the terms of lower orders may not be neglected in this case. This becomes apparent if we examine the cofactor of the difference product:

$$|\cos(jx_i)|_n = \begin{vmatrix} 0 & 1 & 0 & 0 & \cdots & \cdots & 0 \\ -1 & 0 & 2 & 0 & & & 0 \\ 0 & -3 & 0 & 4 & 0 & & 0 \\ 1 & 0 & -8 & 0 & 8 & \ddots & \vdots \\ & \ddots & \ddots & & \ddots & \ddots & 0 \\ \ddots & \cdots & \sum_{l=4}^{\lfloor \frac{n}{2} \rfloor} \binom{l}{4}\binom{n}{2l} & 0 & \sum_{l=2}^{\lfloor \frac{n}{2} \rfloor} \binom{l}{2}\binom{n}{2l} & 0 & 2^{n-1} \\ S_n & S_{n-1} & S_{n-2} & S_{n-3} & \cdots & S_1 & S_0 \end{vmatrix} \mathrm{DP}(\cos(x_1), \ldots, \cos(x_n)).$$

This cofactor determinant has a diagonal band form and we see that it is not possible to get rid of the nonzero first upper side diagonal for symbolic orders. More complicated cases like $|\cos((j+d)x_i)|$ result in a cofactor determinant with even more nonzero upper diagonals, so we are only able to derive an explicit determinant formula for the cases $|\cos((j-1)x_i)|$ and $|\sin((j-1)x_i)|$ or $|\sin(jx_i)|$ that can be transformed into the first case.

3.2 Double Alternants

A determinant may evidently be an alternant with respect to two sets of variables, the exchange of any two of the one set being equivalent to an exchange of rows, and of any two of the other to an exchange of columns.

Definition 17 *A determinant is called* double alternant *if it is alternating with respect to two sets of variables, $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_n\}$.*

If the generating function is $F(x_i, y_j)_{1 \le i,j \le n}$, we will denote a double alternant DA by

$$DA(F(x_i, y_j)) = DA(F(x_1, y_1), F(x_2, y_2), \ldots, F(x_n, y_n)) = \begin{vmatrix} F(x_1,y_1) & F(x_1,y_2) & \cdots & F(x_1,y_n) \\ F(x_2,y_1) & F(x_2,y_2) & \cdots & F(x_2,y_n) \\ \vdots & \vdots & & \vdots \\ F(x_n,y_1) & F(x_n,y_2) & \cdots & F(x_n,y_n) \end{vmatrix}.$$

A very well known example for double alternants is *Cauchy's double alternant*, which is generated by the function $F(x_i, y_j) = \frac{1}{x_i + y_j}$, $x_i + y_j \neq 0$ for $1 \le i, j \le n$. Hence, we have

$$CauchyDA = \begin{vmatrix} \frac{1}{x_1+y_1} & \frac{1}{x_1+y_2} & \cdots & \frac{1}{x_1+y_n} \\ \frac{1}{x_2+y_1} & \frac{1}{x_2+y_2} & \cdots & \frac{1}{x_2+y_n} \\ \vdots & \vdots & & \vdots \\ \frac{1}{x_n+y_1} & \frac{1}{x_n+y_2} & \cdots & \frac{1}{x_n+y_n} \end{vmatrix}.$$

It is straightforward to see that any double alternant has the two difference products of the two variable sets as a factor since substituting one variable of the first set for another variable of the first set yields two identical rows and substituting one variable from the second set for another of the second set in two identical columns. Hence we have the following theorem:

Theorem 16 *Any double alternant DA with entries in the ring of polynomials $R[x_1, \ldots, x_n][y_1, \ldots, y_n]$ has the difference product of its variable sets as a factor, i.e., considering the variable sets $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_n\}$, we have*

$$DA = S \cdot DP(x_1, \dots, x_n)\, DP(y_1, \dots, y_n),$$

for a function S which is symmetric with respect to both sets of variables.

It is possible to modify Theorem 15 to get a similar result for double alternants which enables us to express double alternants in terms of combinations of elementary symmetric functions of the two variable sets.

Theorem 17 *Consider a double alternant of order n generated by $F(x_i, y_j) = \sum_{0 \le \kappa, \lambda \le r} a_{\kappa\lambda} x_i^\kappa y_j^\lambda$ and let $S_k = (-1)^k \sigma_k(x_1, \dots, x_n)$ and $S'_k = (-1)^k \sigma_k(y_1, \dots, y_n)$. The following identity holds:*

$$\frac{|F(x_1, y_1), \dots, F(x_n, y_n)|}{DP(x_1, \dots, x_n) DP(y_1, \dots, y_n)} =$$

$$(-1)^{r-n+1} \left| \begin{array}{ccccccccccc}
a_{00} & a_{10} & \cdots & a_{n0} & a_{n+1,0} & \cdots & a_{r0} & S'_n & 0 & \cdots & 0 \\
a_{01} & a_{11} & \cdots & a_{n1} & a_{n+1,1} & \cdots & a_{r1} & S'_{n-1} & S'_n & \ddots & \vdots \\
\vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots & S'_{n-1} & \ddots & 0 \\
a_{0n} & a_{1n} & \cdots & a_{nn} & a_{n+1,n} & \cdots & a_{rn} & S'_0 & \vdots & \ddots & S'_n \\
a_{0,n+1} & a_{1,n+1} & \cdots & a_{n,n+1} & a_{n+1,n+1} & \cdots & a_{rn+1} & 0 & S'_0 & & S'_{n-1} \\
\vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots & \ddots & \ddots & \vdots \\
a_{0r} & a_{1r} & \cdots & a_{nr} & a_{n+1,r} & \cdots & a_{rr} & 0 & \cdots & 0 & S'_0 \\
S_n & S_{n-1} & \cdots & S_0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\
0 & S_n & S_{n-1} & \cdots & S_0 & \ddots & \vdots & \vdots & & \vdots & \vdots \\
\vdots & \ddots & \ddots & \ddots & & \ddots & 0 & 0 & \cdots & 0 & 0 \\
0 & \cdots & 0 & S_n & S_{n-1} & \cdots & S_0 & 0 & \cdots & 0 & 0
\end{array} \right|_{2(r+1)-n},$$

Note that the band sub matrices of the S_i and the S'_i have dimension $(r-n+1) \times (r+1)$ (respectively other way round), such that we get a determinant involving only coefficients if $r = n-1$, as we expect.

Proof.

The proof is similar to the proof of Theorem 15, the starting point being the multiplication of the right–hand determinant by the two determinants $\mathrm{DP}(x_1, \dots, x_n, \omega_0, \dots, \omega_{r-n})$ and $\mathrm{DP}(y_1, \dots, y_n, \pi_0, \dots, \pi_{r-n})$.

Let us denote the right–hand determinant with Δ again. We try to multiply Δ from the left with

$\mathrm{DP}(y_1, \dots, y_n, \pi_0, \dots, \pi_{r-n})$ and from the right with $\mathrm{DP}(x_1, \dots, x_n, \omega_0, \dots, \omega_{r-n})$. To be able to multiply these determinants we have to achieve the same determinant orders, so we stretch the difference products by an identity matrix. First, we consider $\mathrm{DP}(y_1, \dots, y_n, \pi_0, \dots, \pi_{r-n}) \cdot \Delta$:

$$\left| \begin{array}{ccccccccc}
1 & y_1 & \cdots & y_1^n & \cdots & y_1^r & 0 & \cdots & 0 \\
\vdots & \vdots & & \vdots & & \vdots & \vdots & & \vdots \\
1 & y_n & \cdots & y_n^n & \cdots & y_n^r & 0 & \cdots & 0 \\
1 & \pi_0 & \cdots & \pi_0^n & \cdots & \pi_0^r & 0 & \cdots & 0 \\
\vdots & \vdots & & \vdots & & \vdots & \vdots & & \vdots \\
1 & \pi_{r-n} & \cdots & \pi_{r-n}^n & \cdots & \pi_{r-n}^r & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & \cdots & 0 & 1 & \ddots & \vdots \\
\vdots & \vdots & & \vdots & & \vdots & \ddots & \ddots & 0 \\
0 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 & 1
\end{array} \right| \cdot \left| \begin{array}{cccccccc}
a_{00} & \cdots & a_{n0} & \cdots & a_{r0} & S'_n & & \mathbf{0} \\
\vdots & & \vdots & & \vdots & \vdots & \ddots & \\
a_{0n} & \cdots & a_{nn} & \cdots & a_{rn} & S'_n & & S'_0 \\
\vdots & & \vdots & & \vdots & 0 & \ddots & \vdots \\
a_{0r} & \cdots & a_{nr} & \cdots & a_{rr} & \vdots & \ddots & S_{0'} \\
S_n & \cdots & S_0 & 0 & \cdots & 0 & \cdots & 0 \\
 & \ddots & & \ddots & \ddots & \vdots & & \vdots \\
\mathbf{0} & & S_n & \cdots & S_0 & 0 & \cdots & 0
\end{array} \right|$$

$$
= \left|\begin{array}{ccccccccc}
\sum_{k=0}^r a_{0k}y_1^k & \cdots & \sum_{k=0}^r a_{nk}y_1^k & \cdots & \sum_{k=0}^r a_{rk}y_1^k & \psi(y_1) & y_1\psi(y_1) & \cdots & y_1^{r-n}\psi(y_1) \\
\vdots & & \vdots & & \vdots & \vdots & \vdots & & \vdots \\
\sum_{k=0}^r a_{0k}y_n^k & \cdots & \sum_{k=0}^r a_{nk}y_n^k & \cdots & \sum_{k=0}^r a_{rk}y_n^k & \psi(y_n) & y_n\psi(y_n) & \cdots & y_n^{r-n}\psi(y_n) \\
\sum_{k=0}^r a_{0k}\pi_0^k & \cdots & \sum_{k=0}^r a_{nk}\pi_0^k & \cdots & \sum_{k=0}^n a_{rk}\pi_0^k & \psi(\pi_0) & \pi_0\psi(\pi_0) & \cdots & \pi_0^{r-n}\psi(\pi_0) \\
\vdots & & \vdots & & \vdots & \vdots & \vdots & & \vdots \\
\sum_{k=0}^r a_{ok}\pi_{r-n}^k & \cdots & \sum_{k=0}^r a_{nk}\pi_{r-n}^k & \cdots & \sum_{k=0}^r a_{rk}\pi_{r-n}^k & \psi(\pi_{r-n}) & \pi_{r-n}\psi(\pi_{r-n}) & \cdots & \pi_{r-n}^{r-n}\psi(\pi_{r-n}) \\
S_n & \cdots & S_0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
 & \ddots & & \ddots & \ddots & \vdots & \vdots & & \vdots \\
\mathbf{0} & & S_n & \cdots & S_0 & 0 & 0 & \cdots & 0
\end{array}\right|,
$$

where $\psi(y) = S_0'y^n + S_1'y^{n-1} + \cdots + S_{n-1}'y + S_n' = (y-y_1)\cdots(y-y_n)$.

Observe that $\psi(y_1) = \cdots = \psi(y_n) = 0$, thus we have a zero block in the upper right corner. To get a similar simplification as in Theorem 15, we have to swap the $(r-n+1)\times(2(r+1)-n)$ block containing the S_i's with the block containing $\pi_0,\ldots,\pi_n$ of same block size:

$$
= (-1)^{r-n+1}\left|\begin{array}{cccccccc}
\sum_{k=0}^r a_{0k}y_1^k & \cdots & \sum_{k=0}^r a_{nk}y_1^k & \cdots & \sum_{k=0}^r a_{rk}y_1^k & 0 & \cdots & 0 \\
\vdots & & \vdots & & \vdots & \vdots & & \vdots \\
\sum_{k=0}^r a_{0k}y_n^k & \cdots & \sum_{k=0}^r a_{nk}y_n^k & \cdots & \sum_{k=0}^r a_{rk}y_n^k & 0 & \cdots & 0 \\
S_n & \cdots & S_0 & 0 & \cdots & 0 & \cdots & 0 \\
 & \ddots & & \ddots & \ddots & \vdots & & \vdots \\
\mathbf{0} & & S_n & \cdots & S_0 & 0 & \cdots & 0 \\
\sum_{k=0}^r a_{0k}\pi_0^k & \cdots & \sum_{k=0}^k a_{nk}\pi_0^k & \cdots & \sum_{k=0}^r a_{rk}\pi_0^k & \psi(\pi_0) & \cdots & \pi_0^{r-n}\psi(\pi_0) \\
\vdots & & \vdots & & \vdots & \vdots & & \vdots \\
\sum_{k=0}^r a_{0k}\pi_{r-n}^k & \cdots & \sum_{k=0}^r a_{nk}\pi_{r-n}^k & \cdots & \sum_{k=0}^r a_{rk}\pi_{r-n}^k & \psi(\pi_{r-n}) & \cdots & \pi_{r-n}^{r-n}\psi(\pi_{r-n})
\end{array}\right|,
$$

Now we will multiply with the "stretched" determinant

$$
\mathrm{DP}(x_1,\ldots,x_n,\omega_0,\ldots,\omega_{r-n}) = \left|\begin{array}{ccccccccc}
1 & \cdots & 1 & 1 & \cdots & 1 & 0 & \cdots & 0 \\
x_1 & \cdots & x_n & \omega_0 & \cdots & \omega_{r-n} & 0 & \cdots & 0 \\
\vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\
x_1^n & \cdots & x_n^n & \omega_0^n & \cdots & \omega_{r-n}^n & 0 & \cdots & 0 \\
\vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\
x_1^r & \cdots & x_n^{r-n} & \omega_0^{r-n} & \cdots & \omega_{r-n}^{r-n} & 0 & \cdots & 0 \\
0 & \cdots & 0 & 0 & \cdots & 0 & 1 & \ddots & \vdots \\
\vdots & & \vdots & \vdots & & \vdots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & 0 & 1
\end{array}\right|
$$

from the right and get

$$= (-1)^{r-n+1} \begin{vmatrix} F(x_1,y_1) & \cdots & F(x_n,y_1) & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ F(x_1,y_n) & \cdots & F(x_n,y_n) & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \phi(x_1) & \cdots & \phi(x_n) & \phi(\omega_0) & \cdots & \phi(\omega_{r-n}) & 0 & \cdots & 0 \\ \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ x_1^{r-n}\phi(x_1) & \cdots & x_n^{r-n}\phi(x_n) & \omega_0^{r-n}\phi(\omega_0) & \cdots & \omega_{r-n}^{r-n}\phi(\omega_{r-n}) & 0 & \cdots & 0 \\ * & \cdots & * & * & \cdots & * & \psi(\pi_0) & \cdots & \pi_0^{r-n}\psi(\pi_0) \\ \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ * & \cdots & * & * & \cdots & * & \psi(\pi_{r-n}) & \cdots & \pi_{r-n}^{r-n}\psi(\pi_{r-n}) \end{vmatrix},$$

where $\phi(x) = S_0x^n + S_1x^{n-1} + \cdots + S_{n-1}x + S_n = (x - x_1)\cdots(x - x_n)$. Making use of the block structure, we obtain

$$= (-1)^{r-n+1} \begin{vmatrix} F(x_1,y_1) & \cdots & F(x_1,y_n) \\ \vdots & & \vdots \\ F(x_n,y_1) & \cdots & F(x_n,y_n) \end{vmatrix} \begin{vmatrix} \phi(\omega_0) & \cdots & \phi(\omega_{r-n}) \\ \vdots & & \vdots \\ \omega_0^{r-n}\phi(\omega_0) & \cdots & \omega_{r-n}^{r-n}\phi(\omega_{r-n}) \end{vmatrix} \begin{vmatrix} \psi(\pi_0) & \cdots & \pi_0^{r-n}\psi(\pi_0) \\ \vdots & & \vdots \\ \psi(\pi_{r-n}) & \cdots & \pi_{r-n}^{r-n}\psi(\pi_{r-n}) \end{vmatrix}.$$

Removing common factors with $\mathrm{DP}(x_1,\ldots,x_n,\omega_0,\ldots,\omega_{r-n})$ and $\mathrm{DP}(y_1,\ldots,y_n,\pi_0,\ldots,\pi_{r-n})$ like in Theorem 15 indeed yields the desired identity.

$\square$

Let us see if we can also benefit from this theorem.

Example

Consider the double alternant DA of order n, generated by $F(x_i,y_j) = (x_i + y_j)^n = \sum_{k=0}^{n} \binom{n}{k} x_i^k y_j^{n-k}$. Theorem 17 gives

$$\frac{DA}{\mathrm{DP}(x_1,\ldots,x_n)\mathrm{DP}(y_1,\ldots,y_n)} = - \begin{vmatrix} 0 & \cdots & \cdots & 0 & \binom{n}{n} & S'_n \\ \vdots & & \cdot & \binom{n}{n-1} & 0 & S'_{n-1} \\ \vdots & \cdot & \cdot & \cdot & \vdots & \vdots \\ 0 & \binom{n}{1} & \cdot & & \vdots & S'_1 \\ \binom{n}{0} & 0 & \cdots & \cdots & 0 & S'_0 \\ S_n & S_{n-1} & \cdots & S_1 & S_0 & 0 \end{vmatrix}_{n+2}.$$

Examining the right–hand determinant, we observe that it is possible to transform it into arrow form. Applying the results of the second chapter, we get the formula

$$(-1)^{\lfloor \frac{n+2}{2} \rfloor} \sum_{l=0}^{n} \sigma_l(x_1,\ldots,x_n)\sigma_{n-l}(y_1,\ldots,y_n) \prod_{\substack{k=0 \\ k\neq l}}^{n} \binom{n}{k}.$$

In fact it is possible to reach frame form for other special cases: Examining $F(x_i,y_j) = \sum_{k=0}^{deg} \sum_{l=1}^{r} c_l x_i^{p_l} y_j^{p_l}$, we observe for constant d that $x_i^d y_j^k$ is introducing a column, $x_i^k y_j^d$ a row, $x_i^k y_j^k$ a main diagonal and $x_i^k y_j^{n-k}$ a counter main diagonal. If at most two rows, two columns, and one diagonal occur in the cofactor determinant then we may apply the results from the second chapter to compute the determinant formula. Unfortunately this is only possible for a maximum degree $n-1$ or n (of course the whole determinant vanishes for a maximum degree less than $n-1$), since otherwise we have too much rows and columns resulting from the elementary symmetric functions.

However, Theorem 17 is restrictive, since it holds only for double alternants which can be generated by a polynomial function $F(x_i, y_j)$. For example this is not true for Cauchy's double alternant. We will try to derive its formula in the following:

Cauchy's double alternant is generated by a fraction. Following our rules of thumb, we clear the fractions by multiplying with $\prod_{k,l=1}^{n}(x_k + y_l)$. The resulting generating function of the modified determinant is $\hat{F}(x_i, y_j) = \frac{\prod_{l,k=1}^{n}(x_k+y_k)}{(x_i+y_j)}$ which reduces to a polynomial but does not have the form required in the theorem. We know that the modified double alternant has $\mathrm{DP}(x_1, \ldots, x_n)$ $\mathrm{DP}(y_1, \ldots, y_n)$ as a factor. Since both the double alternant and the difference products have the same order, the cofactor is numerical only. If we put $x_i = -y_i$, all the elements vanish, except those in the main diagonal and we obtain $\mathrm{DP}(x_1, \ldots, x_n)$. Hence, the cofactor has to be 1 and the formula of Cauchy's Double Alternant is

$$CauchyDA(x_i, y_j) = \frac{\mathrm{DP}(x_1, \ldots, x_n)\mathrm{DP}(y_1, \ldots, y_n)}{\prod_{i,j=1}^{n}(x_i + y_j)}.$$

3.3 Implementation

In the previous sections, we derived several methods to compute determinant formulas for different types of alternants. We will briefly describe the implementation of a Maple package containing most of these algorithms. Examples and further details can be found in the appendix or the on–line help pages.

3.3.1 General Considerations

Matrix order Like in the FRAMEFORMS package, the matrix order n can either be a positive integer or a symbolic value of the form $n+d$, d integer. The first case results in a determinant which could also be computed by Maple's `det` function. For $n > 5$ it will be worth to use the package, since either direct computation is provided or the determinant of the alternant is computed via the determinant of simpler cofactor matrices such that we do not get the enormous expression swell of intermediate polynomial computations.

Specifying the matrix Alternants may be specified with a single polynomial function $f(x_i, j)$ generating the elements a_{ij}. Alternatively, a piecewise specification (like in the FRAMEFORMS package) is possible, in order to have different column generating functions: `[ [1..p1, f1], [p1+1..p2,f2] , ... , [n-pk..n,fk] ]`.

Checking the determinant The check facility for the computed determinant formulas is identical to the FRAMEFORMS package. It is worth to note, however, that large check values (even values of 6 and 7) can cause Maple to run out of memory since polynomial entries lead to gigantic intermediate expressions.

3.3.2 The package functions

The ALTERNANT package consists of the following functions:

`Alternant, AlternantMatrix, ESFcofactorMatrix, CSFcofactorMatrix, DP, ESF, S_to_esf, evalesf, CSF, evalcsf, DoubleAlternant, DoubleAlternantMatrix, DoubleAlternantCofactorMatrix, TrigAlternant.`

Alternant The function `Alternant` computes the determinant of the specified alternant for a maximum degree of $n+d$ of the generating polynomials.

The input is tested for validity and the generating functions are transformed into a special list form, e.g. $cx_i^j - x_i^3$ is represented as `[[c,j],[-1,3]]`, to simplify occurring computations. Then it is decided if a formula can be computed and which strategy to use. We distinguish three different strategies: `normal`, `esf` and `csf` that can be enforced using an optional directive.

- If the directive `normal` is chosen then we require a total function $f(x_i, j)$ defining the alternant. We parse f to see if we can apply Corollary 10 or try to use Corollary 11 instead.
- An alternant formula using complete symmetric functions can only be computed if the first $n-p$ columns are generated by a monomial x_i^{j+d}. In that case we pull out factors and determine the resulting cofactor determinant which is upper triangular for the first $n-p$ columns. Thus, we simulate expansion until we reach the $p \times p$ minor that is computed using Maple's `det` function. If the generating functions for the last p columns are polynomials we split the computation into different monomial alternants and proceed as before. The returned formula is the cofactor determinant formula multiplied with the difference product of the variables.
- A formula involving elementary symmetric functions can be computed if the columns pos_1 to $n-pos_2$ are generated by a monomial x_i^{j+d} or if we have a single generating function of the form $c_1 x_i^{j+p_1} + c_2 x_i^{j+p_2}$ or $c_l x_i^{j+p_1} + \sum_{l=2}^{k} c_l x_i^{p_l}$. In the first case we split the computation into the computation of monomial alternants. We determine their cofactor determinant (first rearranging columns in order to have increasing powers) and expand it (recall that monomial alternants have only one nonzero element in each of the first n rows). For each of the first n rows we delete the row and column containing the nonzero element (the determinant vanishes if no nonzero element exists or another nonzero element occurs in the same column). Since we only allowed one generating monomial between the columns pos_1 to $n-pos_2$ it is possible to simulate the deletion of an "infinite" number of rows and columns. Hence we are left with a fixed integer order determinant containing elementary symmetric functions that is computed using Maple's `det` function. In the other cases we proceed as described in subsection 3.1.5, first eliminating all but one constant columns (or the second diagonal) and afterwards expanding the determinant taking the modifications into account.

AlternantMatrix This function returns the matrix of the specified alternant. It is useful for illustrative purposes and as a tool to play with. In alternant matrices of symbolic order we use dots "o" to simulate the symbolic order which implies that it should be used for illustration only since Maple treats them as usual matrix entries.

ESFcofactorMatrix This function returns the matrix of the cofactor involving elementary symmetric functions of the difference product of the specified alternant (see Theorem 15). The elementary symmetric functions of order k are denoted as `S(k,n,x)` to preserve readability. As above, dots are used to illustrate symbolic orders, hence the same comments apply.

CSFcofactorMatrix This function returns the matrix of the cofactor involving complete symmetric functions of the difference product of the specified alternant (see Theorem 14). As above, dots are used to illustrate symbolic orders, hence the same comments apply.

DP The function `DP(n,x,f)` computes the difference product of the input function $f(x_i)$. For $f = x_i$ we get the difference product of the variables.

ESF The function `ESF(k,n,x)` computes the kth elementary symmetric function of the variables $x_1, \ldots, x_n$ (for $0 \le k \le n$ integer) according to Definition 15.

S_to_esf The function `S_to_esf(expr)` substitutes each unevaluated occurrence of `S(k,n,x)` by $(-1)^k\cdot$`esf(k,n,x)`. We used the unevaluated `S(k,n,x)` calls as abbreviation to preserve readability. The resulting expression can be fully evaluated for integer n using the function `evalesf`.

evalesf The functions `evalesf(expr)` evaluates an expression containing unevaluated `esf` calls that were used to preserve readability of the determinant formulas.

CSF The function `CSF(k,n,x)` computes the kth complete symmetric function of the variables $x_1, \ldots, x_n$ (for k, n integer) using the recurrence (3.2).

evalcsf The functions `evalcsf(expr)` evaluates an expression containing unevaluated `csf` calls that were used to preserve readability of the determinant formulas.

DoubleAlternant The function `DoubleAlternant` computes the determinant formula of a special form of double alternants. For integer orders we only require the input generating function to be a polynomial in x_i and y_j. We determine the matrix of the cofactor and compute the determinant of this generally sparse matrix with simpler entries (integers, integer variables and elementary symmetric polynomials) than the original double alternant. For symbolic double alternants, we have the following restrictions:

$$F(x_i, y_j) = (ax_i + by_j)^{deg} \text{ or } F(x_i, y_j) = \sum_{k=0}^{deg} \sum_{l=1}^{m} c_l x_i^{p_l} y_j^{q_l}.$$

If the input generating functions meets these restrictions, we try to detect frame form, observing that $x_i^k y_j^k$ yields a diagonal, $x_i^k y_j^{n-k}$ yields a counter diagonal, $x_i^k y_j^d$, (d integer) yields a row, $x_i^d y_j^k$ yields a column and obtain a formula using the FRAMEFORMS package in that case. No formula can be obtained for a maximum degree bigger than n or other generating formulas not meeting these restrictions.

DoubleAlternantMatrix This function returns the matrix of the specified double alternant. As above, dots are used to illustrate symbolic orders, hence the same comments apply.

DoubelAlternantCofactorMatrix This function returns the matrix of the cofactor of the difference products of the specified double alternant according to Theorem 17. As above, dots are used to illustrate symbolic orders, hence the same comments apply.

TrigAlternant The function `TrigAlternant` computes the determinant for a very restricted class of trigonometric alternants. The generating functions have to be of the form

$$sin((j-k)x_i + d) \quad k \geq 0,$$

or

$$cos((j-k)x_i + d) \quad k \geq 1.$$

The computation follows the method of the example in subsection 3.1.6.

3.3.3 Problems and Restrictions

Obviously, there are several special cases of alternants that are not included in the package. However, it is very cumbersome to include more and more special cases in the package and at a certain point, the end just does not certify the means. Hence, we restricted ourselves to implement all the general methods.

Like in the FRAMEFORMS package, we have the problems that the simplified output formulas are sometimes far from simplified in our view.

3.4 Summary

This chapter dealt with the important class of alternants. We showed that a polynomial alternant has the difference product of its variables as a factor, the cofactor being a symmetric function. We presented two early results from [Met60] expressing the symmetric cofactor as a determinant with elementary symmetric polynomials and complete symmetric polynomials respectively. For simple alternants of degree $n + k$ it was possible to compute an explicit formula for the cofactor since the cofactor determinants had a simple structure. The multilinearity of the determinant allowed to generalize the results for matrices with finitely many polynomial entries. However, apart from a few special cases we could not derive explicit formulas for alternants with polynomial entries of arbitrary degree.

The problem of non–polynomial entries was briefly discussed. We described how it is possible to transform a certain class of trigonometric alternants into easily structured polynomial alternants. Explicit formulas were obtained in a few special cases only.

In the last section, we investigated double alternants, i.e. functions that alternate between two sets of variables. A similar result from [Met60] was proven that expressed the symmetric cofactor as a determinant containing coefficients and the elementary symmetric polynomials of the two variable sets. It was possible to derive a formula for some special cases for which the cofactor determinant could be transformed into frameform and techniques of the first chapter could be applied. The presented theorem was more restrictive than the one concerning alternants, e.g. it was not possible to compute the formula of Cauchy's double alternant using the theorem.

The implemented Maple package is able to handle almost all covered special cases of alternants and double alternants. Using the elementary or complete symmetric function representation, the computed determinant formulas are even quite "readable" (see the appendix for examples). For integer alternant orders larger than 6 it may even be necessary to use the package functions to obtain results, since the normal `det` function may collapse due to the size of intermediate expressions.

Chapter 4

Determinants of Hessenberg and tridiagonal matrices

In this chapter we will investigate determinants of matrices which only have certain diagonals nonzero. Important matrix classes of this type are the tridiagonal matrices and the Hessenberg matrices.

4.1 Continuants or Determinants of tridiagonal matrices

Tridiagonal matrices occur in many application areas, like spline interpolation, symmetric eigenvalue problems and differential equations. We will derive a general determinant formula for tridiagonal matrices and examine its structure to relate this problem to interesting other problems. Our presentation follows [Met60] and [GKP92].

Definition 18 *A* continuant *is the determinant of a tridiagonal matrix, i.e. a matrix which is zero except the main diagonal and the two adjacent side diagonals.*

The continuant generated by the functions $md : \{1, \dots, n\} \to R$, $md(i) = a_i$ *for the main diagonal,* $ud : \{1, \dots, n-1\} \to R$, $ud(i) = b_i$ *for the first upper side diagonal and* $ld : \{1, \dots, n-1\} \to R$, $ld(i) = c_i$ *for the first lower side diagonal will be denoted as follows:*

$$K_{a_i,b_i,c_i}(1, \dots, n) = \begin{vmatrix} a_1 & b_1 & 0 & \cdots & 0 \\ c_1 & a_2 & b_2 & \ddots & \vdots \\ 0 & c_2 & a_3 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & b_{n-1} \\ 0 & \cdots & 0 & c_{n-1} & a_n \end{vmatrix}.$$

$K_{a_i,b_i,c_i}(1, \dots, n)$ *is abbreviated by* $K(1, \dots, n)$ *when the diagonal generating functions are known.*

Let $K_{a_i,b_i,c_i}(p_1, \dots, p_2)$ *denote a continuant of order* $p_2 - p_1 + 1$ *with diagonal generating functions* $md : \{p_1, \dots, p_2\} \to R$, $md(i-p_1+1) = a_i$ *for the main diagonal,* $ud : \{p_1, \dots, p_2 - 1\} \to R$, $ud(i-p_1+1) = b_i$ *for the upper side diagonal and* $ld : \{p_1, \dots, p_2 - 1\} \to R$, $ld(i - p_1 + 1) = c_i$ *for the lower side diagonal.*

How do we compute a determinant formula for continuants? The sparseness of the tridiagonal matrices suggests minor expansion and we will see in the following that we can straightforwardly obtain a simple recurrence for the determinant formula.

Theorem 18 (Law of expansion) *Let $K_{a_i,b_i,c_i}(1,\dots,n)$ be a continuant defined as above. The following recursive identities hold:*

$$K(1,\dots,n) = a_1 K(2,\dots,n) - b_1 c_1 K(3,\dots,n), \tag{4.1}$$

and

$$K(1,\dots,n) = a_n K(1,\dots,n-1) - b_{n-1} c_{n-1} K(1,\dots,n-2), \tag{4.2}$$

with $K_{a_i,b_i,c_i}(1) = a_1$, $K_{a_i,b_i,c_i}(0) = 1$.

Proof.

We will prove identity (4.2) by expansion of the last column. We get

$$\begin{aligned}
K_{a_i,b_i,c_i}(1,\dots,n) &= \begin{vmatrix} a_1 & b_1 & 0 & \cdots & 0 \\ c_1 & a_2 & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & b_{n-2} & 0 \\ \vdots & \ddots & c_{n-2} & a_{n-1} & b_{n-1} \\ 0 & \cdots & 0 & c_{n-1} & a_n \end{vmatrix} \\
&= a_n \begin{vmatrix} a_1 & b_1 & 0 & \cdots & 0 \\ c_1 & a_2 & b_2 & \ddots & \vdots \\ 0 & c_2 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & b_{n-2} \\ 0 & \cdots & 0 & c_{n-2} & a_{n-1} \end{vmatrix} - b_{n-1} \begin{vmatrix} a_1 & b_1 & 0 & \cdots & 0 & 0 \\ c_1 & a_2 & \ddots & \ddots & \vdots & \vdots \\ 0 & \ddots & \ddots & b_{n-4} & 0 & 0 \\ \vdots & \ddots & c_{n-4} & a_{n-3} & b_{n-3} & 0 \\ 0 & \cdots & 0 & c_{n-3} & a_{n-2} & b_{n-2} \\ 0 & \cdots & 0 & 0 & 0 & c_{n-1} \end{vmatrix}.
\end{aligned}$$

The determinant in the first term of the righthand side is obviously $K(1,\dots,n-1)$ whereas the determinant in the second term reduces to $K(1,\dots,n-2)$ if we expand the last row, and the identity is established.

Identity (4.1) can be obtained analogously.

□

Example

$$K_{a_i,b_i,c_i}(1,\dots,4) = a_1a_2a_3a_4 - b_1c_1a_3a_4 - a_1b_2c_2a_4 - a_1a_2b_3c_3 + b_1c_1b_3c_3.$$

Let us take a look at the terms in the determinant formula of a continuant $K_{a_i,b_i,c_i}(1,\dots,n)$. One term of the continuant is obviously $a_1a_2\cdots a_n$. Other terms can be obtained from $a_1a_2\cdots a_n$ by replacing any pair of consecutive a's, a_ra_{r+1} by $-b_rc_r$ for $1 \le r \le n-1$. This follows from the definition and from the fact that to get b_r and c_r in the position of a_r and a_{r+1}, one row or column exchange is necessary. Of course it is possible to iterate this process on the terms obtained by this method which allows us to successively derive all the continuant terms.

It is possible to conclude the following observation.

Observation

$$K_{a_i,b_i,c_i}(1,\dots,n) = K_{a_i,-1,-b_ic_i}(1,\dots,n) = K_{a_i,1,b_ic_i}(1,\dots,n). \tag{4.3}$$

It follows that every term in a continuant of odd order must contain a main diagonal element, whereas in a continuant of even order, we have the term $(-1)^{\frac{n}{2}}b_1c_1b_3c_3\cdots b_{n-1}c_{n-1}$ which contains no element of the main diagonal and is obtained by consecutively replacing a_ra_{r+1} by $-b_rc_r$ starting with $a_1a_2\cdots a_n$ and $r=1$.

Hence, we have the following corollary.

Corollary 12 *Let $K_{0,b_i,c_i}(1,\dots,n)$ be a continuant with zero main diagonal.*

1. *If n is odd then $K_{0,b_i,c_i}(1,\dots,n)=0$.*
2. *If n is even then $K_{0,b_i,c_i}(1,\dots,n)=(-1)^{\frac{n}{2}}b_1c_1b_3c_3\cdots b_{n-1}c_{n-1}$.*

Now let us examine a special class of continuants to gain more insight into the structure of continuant polynomials. We assume that the side diagonals are 1 and -1 respectively and will write $K(x_1,\dots,x_n)$ for $K_{x_i,1,-1}(1,\dots,n)$. Thus we have the recurrence $K(x_1,\dots,x_n) = x_nK(x_1,\dots,x_{n-1}) + K(x_1,\dots,x_{n-2})$ with $K(x_1)=x_1$ and $K()=1$.

It is easy to see that the number of terms in $K(x_1,\dots,x_n)$ is a Fibonacci number:

$$K(1,\dots,1) = F_{n+1.}$$

Euler observed (like we stated above for a more general case) that $K(x_1,\dots,x_n)$ can be obtained by starting with the product $x_1x_2\cdots x_n$ and then striking out adjacent pairs x_kx_{k+1} in all possible ways. We can represent Euler's rule graphically by constructing all "Morse code" sequences of dots and dashes having length n, where each dot contributes 1 to the length and each dash contributes 2. Here are the Morse code sequences of length 4: $\cdot\cdot\cdot\cdot$, $\cdot\cdot -$, $\cdot - \cdot$, $- \cdot\cdot$, $--$

These dot–dash patterns correspond to the terms of $K(x_1,x_2,x_3,x_4)$; a dot signifies a variable that is included and a dash a pair of variables that is excluded. For example $\cdot - \cdot$ corresponds to x_1x_4.

A Morse code sequence of length n that has k dashes, has $n-2k$ dots and $n-k$ symbols altogether. These dots and dashes can be arranged in $\binom{n-k}{k}$ ways; therefore if we replace each dot by x and each dash by 1 we get

$$K_n(x,x,\dots,x) = \sum_{k=0}^{\lfloor\frac{n}{2}\rfloor}\binom{n-k}{k}x^{n-2k},$$

since we can strike out at most $\lfloor\frac{n}{2}\rfloor$ adjacent pairs x_kx_{k+1} in $x_1x_2\cdots x_n$.

We have seen that the total number of terms in a continuant is a Fibonacci number; hence we have the following identity

$$F_{n+1} = \sum_{k=0}^{\lfloor\frac{n}{2}\rfloor}\binom{n-k}{k}.$$

It is also possible to get a special formula for the general continuant. If we replace each dot by a and each dash by $-bc$ we obtain

$$K_{a,b,c}(1,\dots,n)=\sum_{k=0}^{\lfloor\frac{n}{2}\rfloor}(-1)^k\binom{n-k}{k}a^{n-2k}b^kc^k. \tag{4.4}$$

We now discuss two applications of continuants following [GKP92]. Continuant polynomials are intimately connected with Euclid's algorithm. Suppose, for example, that the computation of $\gcd(m,n)$ finishes in four steps:

$$\begin{array}{rclllll}
\gcd(m,n) & = & \gcd(n_0,n_1) & n_0 = m, & n_1 & = & n;\\
 & = & \gcd(n_1,n_2) & n_2 = n_0 & \bmod n_1 & = & n_0-q_1n_1;\\
 & = & \gcd(n_2,n_3) & n_3 = n_1 & \bmod n_2 & = & n_1-q_2n_2;\\
 & = & \gcd(n_3,n_4) & n_4 = n_2 & \bmod n_3 & = & n_2-q_3n_3;\\
 & = & \gcd(n_4,0)=n_4. & 0 = n_3 & \bmod n_4 & = & n_3-q_4n_4.
\end{array}$$

Then we have

$$\begin{array}{rclcl}
n_4 & = & n_4 & = & K()n_4;\\
n_3 & = & q_4n_4 & = & K(q_4)n_4;\\
n_2 & = & q_3n_3+n_4 & = & K(q_3,q_4)n_4;\\
n_1 & = & q_2n_2+n_3 & = & K(q_2,q_3,q_4)n_4;\\
n_0 & = & q_1n_1+n_2 & = & K(q_1,q_2,q_3,q_4)n_4.
\end{array}$$

In general, if Euclid's algorithm finds the greatest common divisor d in k steps, after computing the sequence of quotients $q_1,\dots,q_k$, then the starting numbers were $K(q_1,q_2,\dots,q_k)d$ and $K(q_2,q_3,\dots,q_k)d$. This fact was noticed early in the eighteenth century by Thomas Fantet de Lagny. He pointed out that consecutive Fibonacci numbers, which occur as continuants when the q's take their minimum value, are therefore the smallest inputs that cause Euclid's algorithm to take a given number of steps.

Continuants are also intimately connected to continued fractions, from which they get their name. We have, for example,

$$a_1+\frac{1}{a_2+\frac{1}{a_3+\frac{1}{a_4}}}=\frac{K(a_1,a_2,a_3,a_4)}{K(a_2,a_3,a_4)}.$$

The same pattern holds for continued fractions of any depth which allows us to formulate the following theorem.

Theorem 19 *The following formula holds:*

$$a_1+\frac{1}{a_2+\frac{1}{\dots+\frac{1}{a_{n-1}+\frac{1}{a_n}}}}=\frac{K(a_1,\dots,a_n)}{K(a_2,\dots,a_n)}. \tag{4.5}$$

Proof.

We will prove the theorem by induction on n.

Base case: For $n = 1$ we trivially have $\frac{K(a_1)}{K()} = a_1$.

Induction step: For the induction step we need the identity

$$K(x_1, \dots, x_{n-1}, x_n + y) = K(x_1, \dots, x_{n-1}, x_n) + yK(x_1, \dots, x_{n-1}). \tag{4.6}$$

The identity holds, since

$$\begin{aligned} K(x_1, \dots, x_{n-1}, x_n + y) &= (x_n + y)K(x_1, \dots, x_{n-1}) + K(x_1, \dots, x_{n-2}) \\ &= x_n K(x_1, \dots, x_{n-1}) + K(x_1, \dots, x_{n-2}) + yK(x_1, \dots, x_{n-1}) \end{aligned}$$

according to (4.2).

Assume, the following formula holds for all values of n:

$$a_1 + \cfrac{1}{a_2 + \cfrac{1}{\dots + \cfrac{1}{a_{n-2} + \cfrac{1}{a_{n-1}}}}} = \frac{K(a_1, \dots, a_{n-1})}{K(a_2, \dots, a_{n-1})}.$$

We observe that

$$a_1 + \cfrac{1}{a_2 + \cfrac{1}{\dots + \cfrac{1}{a_{n-2} + \cfrac{1}{a_{n-1} + \cfrac{1}{a_n}}}}} = \frac{K(a_1, \dots, a_{n-1} + \frac{1}{a_n})}{K(a_2, \dots, a_{n-1} + \frac{1}{a_n})}$$

using the induction hypothesis and replacing a_{n-1} by $a_{n-1} + \frac{1}{a_n}$. Applying identity (4.6), we get

$$\begin{aligned} \frac{K(a_1, \dots, a_{n-2}, a_{n-1} + \frac{1}{a_n})}{K(a_2, \dots, a_{n-2}, a_{n-1} + \frac{1}{a_n})} &= \frac{K(a_1, \dots, a_{n-1}) + \frac{1}{a_n}K(a_1, \dots, a_{n-2})}{K(a_2, \dots, a_{n-1}) + \frac{1}{a_n}K(a_2, \dots, a_{n-2})} \\ &= \frac{a_n K(a_1, \dots, a_{n-1}) + K(a_1, \dots, a_{n-2})}{a_n K(a_2, \dots, a_{n-1}) + K(a_2, \dots, a_{n-2})} = \frac{K(a_1, \dots, a_n)}{K(a_2, \dots, a_n)}, \end{aligned}$$

which completes the induction step.

$\square$

Note that it is also possible to state the preceding theorem for general continuants. The proof is identical.

Theorem 20 *The following identity holds:*

$$a_1 - \cfrac{b_1 c_1}{a_2 - \cfrac{b_2 c_2}{\dots - \cfrac{b_{n-2} c_{n-2}}{a_{n-1} - \cfrac{b_{n-1} c_{n-1}}{a_n}}}} = \frac{K_{a_i, b_i, c_i}(1, \dots, n)}{K_{a_i, b_i, c_i}(2, \dots, n)}.$$

Now we want to turn to some special continuants. Consider a continuant where each main diagonal element, except the first and the last one, is the sum of the side diagonal elements of the same row.

Theorem 21 *Let K be a continuant of order n given by*

$$K = \begin{vmatrix} d_1+d_2 & d_2 & 0 & \cdots & 0 \\ d_3 & d_3+d_4 & d_4 & \ddots & \vdots \\ 0 & d_5 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & d_{2n-3}+d_{2n-2} & d_{2n-2} \\ 0 & \cdots & 0 & d_{2n-1} & d_{2n-1}+d_{2n} \end{vmatrix}.$$

The following formula holds:

$$K = \sum_{l=0}^{n} d_1 d_3 \cdots d_{2l-1} d_{2l+2} d_{2l+4} \cdots d_{2n}. \tag{4.7}$$

To prove this identity, we will need the following lemma.

Lemma 7 *A continuant of even order $n = 2m$ having the main diagonal elements in the even numbered rows equal, is expressible as a continuant of order m, the relation being:*

$$C_{2m} = \begin{vmatrix} \theta_1 & d_1 & 0 & \cdots & \cdots & \cdots & 0 \\ -1 & \phi & d_2 & \ddots & & & \vdots \\ 0 & -1 & \theta_2 & d_3 & \ddots & & \vdots \\ \vdots & \ddots & -1 & \phi & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & d_{2m-2} & 0 \\ \vdots & & & \ddots & -1 & \theta_m & d_{2m-1} \\ 0 & \cdots & \cdots & \cdots & 0 & -1 & \phi \end{vmatrix}_{2m}$$

$$= \begin{vmatrix} \theta_1\phi & d_1 & 0 & \cdots & 0 \\ d_2 & d_2+\theta_2\phi+d_3 & d_3 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & d_{2m-4} & d_{2m-4}+\theta_{m-1}\phi+d_{2m-3} & d_{2m-3} \\ 0 & \cdots & 0 & d_{2m-2} & d_{2m-2}+\theta_m\phi+d_{2m-1} \end{vmatrix}_m.$$

Proof.

We multiply C_{2m} with ϕ^m in the following determinant form

$$\phi^m = \begin{vmatrix} \phi & 0 & 0 & \cdots & \cdots & \cdots & 0 \\ 1 & 1 & -d_2 & \ddots & & & \vdots \\ 0 & 0 & \phi & 0 & \ddots & & \vdots \\ \vdots & \ddots & 1 & 1 & -d_4 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & 0 & 0 & \phi & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & 1 \end{vmatrix}_{2m}$$

(which is obvious if we expand it alternately by row and column) and get

$$C_{2m}\cdot\phi^m = \begin{vmatrix} \theta_1\phi+d_1 & d_1 & -d_1d_2 & 0 & \cdots & & \cdots & 0 \\ 0 & \phi & 0 & 0 & \ddots & & & \vdots \\ -1 & -1 & d_2+\theta_2\phi+d_3 & d_3 & -d_3d_4 & & \ddots & \vdots \\ 0 & 0 & 0 & \phi & & & & 0 \\ \vdots & \ddots & & & \ddots & & 0 & 0 \\ \vdots & & \ddots & -1 & -1 & & d_{2m-2}+\theta_m\phi+d_{2m-1} & d_{2m-1} \\ 0 & \cdots & \cdots & 0 & 0 & & 0 & \phi \end{vmatrix}_{2m}.$$

Observe that we may expand the even rows of the righthand side, obtaining

$$\phi^m \begin{vmatrix} \theta_1\phi+d_1 & -d_1d_2 & 0 & \cdots & 0 \\ -1 & d_2+\theta_2\phi+d_3 & -d_3d_4 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & d_{2m-4}+\theta_{m-1}\phi+d_{2m-2} & -d_{2m-3}d_{2m-2} \\ 0 & \cdots & 0 & -1 & d_{2m-2}+\theta_m\phi+d_{2m-1} \end{vmatrix}_m.$$

Equation (4.3) then implies the lemma.

$\square$

An immediate consequence of this lemma is the following corollary.

Corollary 13 *A continuant of order $n = 2m+1$ having the maindiagonal elements in the odd numbered rows equal,*

$$C_{2m+1} = \begin{vmatrix} \phi & d_1 & 0 & \cdots & \cdots & 0 \\ -1 & \theta_1 & d_2 & \ddots & & \vdots \\ 0 & -1 & \phi & d_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & -1 & \theta_m & d_{2m} \\ 0 & \cdots & \cdots & 0 & -1 & \phi \end{vmatrix}_{2m+1}$$

is expressible as a continuant of order m:

$$C_{2m+1} = \begin{vmatrix} d_1+\theta_1\phi+d_2 & d_2 & 0 & \cdots & 0 \\ d_3 & d_3+\theta_2\phi+d_4 & d_4 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & d_{2m-3} & d_{2m-3}+\theta_{m-1}\phi+d_{2m-2} & d_{2m-2} \\ 0 & \cdots & 0 & d_{2m-1} & d_{2m-1}+\theta_m\phi+d_{2m} \end{vmatrix}. \tag{4.8}$$

Now we are finally able to prove Theorem 21.

Proof.

Putting $\phi = 1$ and $\theta_1 = \cdots = \theta_m = 0$ in (4.8), we have

$$\Delta = \begin{vmatrix} 1 & d_1 & 0 & \cdots & \cdots & 0 \\ -1 & 0 & d_2 & \ddots & & \vdots \\ 0 & -1 & 1 & d_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & -1 & 0 & d_m \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{vmatrix}_{2m+1} = \begin{vmatrix} d_1+d_2 & d_2 & 0 & \cdots & 0 \\ d_3 & d_3+d_4 & d_4 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & d_{2m-3} & d_{2m-3}+d_{2m-2} & d_{2m-2} \\ 0 & \cdots & 0 & d_{2m-1} & d_{2m-1}+d_{2m} \end{vmatrix}_m .$$

Expanding Δ we see that the only nonzero terms are those having a 1 from the main diagonal and the remaining terms from the side diagonals, and these are just the terms included in the sum on the righthand side of (4.7).

$\square$

Note that the first and the last element of the main diagonal may be arbitrary since we can always write them as $c_{11} = d_1 + d_2$ and $c_{nn} = d_{2m-1} + d_{2m}$ with appropriate d_1 and d_{2m}.

Let us illustrate formula (4.7) in a little example. Consider the continuant

$$C_n = \begin{vmatrix} a_0+a_1 & a_1 & 0 & \cdots & 0 \\ a_1 & a_1+a_2 & a_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & a_{n-2} & a_{n-2}+a_{n-1} & a_{n-1} \\ 0 & \cdots & 0 & a_{n-1} & a_{n-1}+a_n \end{vmatrix}_n ,$$

with identical side diagonals. Since each main diagonal element, except the first and the last one, is the sum of the side diagonal elements of the same row, we may apply our formula and obtain the nice identity

$$C_n = \sum_{l=0}^{n} \prod_{\substack{k=0 \\ k\neq l}}^{n} a_k .$$

Note that it is also useful to transform a continuant using the observation (4.3) such that formula (4.7) is applicable. Consider, for example, the continuant

$$C_n = \begin{vmatrix} 1+x^2 & x & 0 & \cdots & 0 \\ x & 1+x^2 & x & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & x & 1+x^2 & x \\ 0 & \cdots & 0 & x & 1+x^2 \end{vmatrix} ,$$

which may be written as

$$C_n = \begin{vmatrix} 1+x^2 & x^2 & 0 & \cdots & 0 \\ 1 & 1+x^2 & x^2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & 1 & 1+x^2 & x^2 \\ 0 & \cdots & 0 & 1 & 1+x^2 \end{vmatrix}$$

using (4.3). Now, it is apparent that we may apply formula (4.7) to get the identity

$$C_n = 1 + x^2 + x^4 + \cdots + x^{2n}.$$

In closing, we will briefly discuss how to split the computation of a continuant, which sometimes is quite useful if the continuant consists of different simple patterns. Splitting is straightforward since a continuant differs from a block continuant in only two elements.

Corollary 14 *Consider a continuant with generating functions a, b, c:*

$$C(1,\ldots,n) = \begin{vmatrix} a_1 & b_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ c_1 & \ddots & \ddots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \ddots & \ddots & b_{m-1} & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & c_{m-1} & a_m & b_m & 0 & \cdots & 0 \\ 0 & \cdots & 0 & c_m & a_{m+1} & b_{m+1} & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & c_{m+1} & \ddots & \ddots & 0 \\ \vdots & & \vdots & \vdots & \ddots & \ddots & \ddots & b_{n-1} \\ 0 & \cdots & 0 & 0 & \cdots & 0 & c_{n-1} & a_n \end{vmatrix}.$$

then

$$C(1,\ldots,n) = C(1,\ldots,m) \cdot C(m+1,\ldots,n) - b_m c_m \cdot C(1,\ldots,m-1) \cdot C(m+2,\ldots,n).$$

Proof.

Clever expansion of the mth row or column yields the result.

$\square$

4.2 Determinants of Hessenberg matrices

Hessenberg matrices frequently occur in the computation of eigenvalues and eigenvectors of a matrix. A general matrix can be reduced to Hessenberg form using finitely many similarity transformations. Afterwards different methods like the QR–method can be used to compute the eigenvalues. See [GvL96] for details.

Definition 19 *A* Hessenberg matrix *is an upper triangular matrix where the first lower diagonal can also contain nonzero elements. For our purposes, we define a Hessenberg matrix HB of order n via its diagonal* *generating functions $d_{-1}(i) = hb_{i+1,i}$ for the lower diagonal, $d_0(i) = hb_{ii}$ for the main diagonal and $d_k(i) = hb_{i,i+k}$ for the kth upper diagonal ($k <= n-1$).*

$$HB_n(d_{-1}, d_0, d_1, \dots, d_{n-1}) = \begin{pmatrix} d_0(1) & d_1(1) & d_2(1) & \ddots & d_{n-2}(1) & d_{n-1}(1) \\ d_{-1}(1) & d_0(2) & d_1(2) & d_2(2) & \ddots & d_{n-2}(2) \\ 0 & d_{-1}(2) & d_0(3) & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & d_1(n-2) & d_2(n-2) \\ \vdots & & \ddots & d_{-1}(n-2) & d_0(n-1) & d_1(n-1) \\ 0 & \cdots & \cdots & 0 & d_{-1}(n-1) & d_0(n) \end{pmatrix}.$$

Trivially, if the diagonal generating functions $d_k(i) = 0$ for $2 \leq k \leq n-1$, we have a tridiagonal matrix.

We could derive a straightforward recurrence formula for the determinant of tridiagonal matrices and would like to have a similar result for determinants of Hessenberg matrices. Following, we show how to establish a quite involved recurrence formula.

Theorem 22 *Let $H(n) = |HB_n(d_{-1}, d_0, d_1, \dots, d_{n-1})|$ be the determinant of a Hessenberg matrix as defined above. This determinant can be computed via the following recurrence:*

$$H(n) = \sum_{i=0}^{n-1} (-1)^i d_i(n-i) H(n-i-1) \prod_{j=1}^{i} d_{-1}(n-j). \tag{4.9}$$

with the base cases $H(1) = d_0(1), H(0) = 1, H(-s) = 0$ for $s > 0$.

Note that $H(n-i) = |HB_{n-1}(d_{-1}, d_0, \dots, d_{n-i-1})|$.

Proof.

We present an inductive argument to prove the recurrence. The base cases are trivial.

Let us expand the last row of $H(n)$:

$$H(n) = d_0(n)H(n-1) - d_{-1}(n-1) \begin{vmatrix} d_0(1) & d_1(1) & \ddots & d_{n-4}(1) & d_{n-3}(1) & d_{n-1}(1) \\ d_{-1}(1) & d_0(2) & d_1(2) & \ddots & d_{n-4}(2) & d_{n-2}(2) \\ 0 & d_{-1}(2) & d_0(3) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & d_1(n-3) & d_3(n-3) \\ 0 & \cdots & 0 & d_{-1}(n-3) & d_0(n-2) & d_2(n-2) \\ 0 & \cdots & 0 & 0 & d_{-1}(n-2) & d_1(n-1) \end{vmatrix}.$$

Denoting the determinant in the second term with D_1 and expanding the last row of D_1, we get

$$D_1 = d_1(n-1)H(n-2) - d_{-1}(n-2) \begin{vmatrix} d_0(1) & d_1(1) & \ddots & d_{n-5}(1) & d_{n-4}(1) & d_{n-1}(1) \\ d_{-1}(1) & d_0(2) & d_1(2) & \ddots & d_{n-5}(2) & d_{n-2}(2) \\ 0 & d_{-1}(2) & d_0(3) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & d_1(n-4) & d_4(n-4) \\ 0 & \cdots & 0 & d_{-1}(n-4) & d_0(n-3) & d_3(n-3) \\ 0 & \cdots & 0 & 0 & d_{-1}(n-3) & d_2(n-2) \end{vmatrix}.$$

Denoting the determinant in the second term with D_2, we iterate this process and finally obtain

$$\begin{aligned} H(n) = d_0(n)H(n-1) - d_{-1}(n-1)\{d_1(n-1)H(n-2-d_{-1}[d_2(n-2)H(n-3) \\ -d_{-1}(n-3)(\ldots(d_{n-2}(2)H(1) - d_{-1}(1)(d_{n-1}(1)H(0)))\ldots)]\}. \end{aligned}$$

Expanding this expression indeed yields the desired recurrence formula

$$H(n) = \sum_{i=0}^{n-1} (-1)^i d_i(n-i)H(n-i-1) \prod_{j=1}^{i} d_{-1}(n-j).$$

❑

Note that recurrence (4.9) simplifies to our known continuant recurrence formula if we have $d_k(i) = 0$ for $k \geq 2$.

Clearly, this involved recurrence relation does not offer explicit formulas for the general case. However, we will discuss some special cases where only a fixed number of the diagonals d_k are nonzero and see how explicit formulas can be obtained.

Example

Consider the determinant $H(n)$ of the Hessenberg matrix of order n generated by the functions $d_{-1}(i) = c, d_0(i) = a$ and $d_{n-2}(i) = b$. Formula (4.9) simplifies to

$$H(n) = aH(n-1) + (-1)^{n-2}bc^{n-2}H(2).$$

Let us expand this formula, we get $H(n) = a(aH(n-2) + (-1)^{n-2}bc^{n-2}H(1)) + (-1)^{n-2}bcH(2)$. Since $H(2) = a^2$, $H(1) = a$ and $H(n-2)$ is a lower triangular determinant, we obtain the formula

$$H(n) = a^n + (-1)^{n-2}2bc^{n-2}a^2.$$

Let us try to generalize the result of our example. Consider Hessenberg matrices generated by the functions $d_{-1}(i), d_0(i)$ and $d_{n-c}(i), \ldots, d_{n-1}(i)$ with constant $c \geq 1$ (this means that the other diagonals are zero).

We show the formula for the value $c = 3$ to better illustrate the method: Expanding the first levels of the recurrence, we get

$$\begin{aligned} H(n) = d_0(n)\Bigg[d_0(n-1)\left\{d_0(n-2)H(n-3) + (-1)^{n-3}d_{n-3}(1)H(0)\prod_{j=1}^{n-3} d_{-1}(n-j-2)\right\} \\ +(-1)^{n-3}d_{n-3}(2)H(1)\prod_{j=1}^{n-3} d_{-1}(n-j-1) + (-1)^{n-2}d_{n-2}(1)H(0)\prod_{j=1}^{n-1} d_{-1}(n-j-1)\Bigg] \\ +(-1)^{n-3}d_{n-3}(3)H(2)\prod_{j=1}^{n-3} d_{-1}(n-j) + (-1)^{n-2}d_{n-2}(2)H(1)\prod_{j=1}^{n-2} d_{-1}(n-j) + (-1)^{n-1}d_{n-1}(1)H(0)\prod_{j=1}^{n-1}(n-j). \end{aligned}$$

Since $H(n-3) = \prod_{l=1}^{n-3} a_l$ for $n \geq 7$, $H(2) = a_1 a_2$, $H(1) = a_1$ and $H(0) = 1$, this simplifies to

$$\begin{aligned} H(n) = \prod_{k=1}^{n} d_0(k) &+ (-1)^{n-3} \sum_{k=1}^{3} d_{n-3}(k) \prod_{l=1}^{3-k} d_0(n-l+1) \prod_{l=1}^{k-1} d_0(l) \prod_{j=1}^{n-3} d_{-1}(n-j-(3-k)) \\ &+ (-1)^{n-2} \sum_{k=1}^{2} d_{n-2}(k) \prod_{l=1}^{2-k} d_0(n-l+1) \prod_{l=1}^{k-1} d_0(l) \prod_{j=1}^{n-2} (d_{-1}(n-j-(2-k)) \\ &+ (-1)^{n-1} d_{n-1}(1) \prod_{j=1}^{n-1} d_{-1}(n-j). \end{aligned}$$

The generalization is straightforward, hence we have the following corollary:

Corollary 15 *Let $HB_n(d_{-1}, d_0, 0, \ldots, 0, d_{n-c}, \ldots, d_{n-1})$ with constant $c \geq 1$ be an order n Hessenberg matrix with $H(n)$ denoting its determinant. The following formula holds for $n \geq 2c+1$:*

$$H(n) = \prod_{k=1}^{n} d_0(k) + \sum_{s=1}^{c} (-1)^{n-s} \left[\sum_{k=1}^{s} \left\{ d_{n-s}(k) \prod_{l=1}^{s-k} d_0(n-l+1) \prod_{l=1}^{k-1} d_0(l) \prod_{j=1}^{n-s} d_{-1}(n-j-(s-k)) \right\} \right]. \tag{4.10}$$

If all the generating functions are constants we get the simpler formula

$$H(n) = d_0^n + 3(-1)^{n-3} d_0^2 d_{n-3} d_{-1}^{n-3} + 2(-1)^{n-2} d_0 d_{n-2} d_{-1}^{n-2} + (-1)^{n-1} d_{n-1} d_{-1}^{n-1},$$

for $c = 3$, which for an arbitrary constant $c \geq 1$ generalizes to

$$H(n) = d_0^n + \sum_{k=1}^{c} k(-1)^{n-k} d_0^{k-1} d_{-1}^{n-k} d_{n-k}. \tag{4.11}$$

In the previous section, we discussed the effect of a zero main diagonal on the determinant of a tridiagonal matrix. Can we make any statements about Hessenberg determinants with zero main diagonal?

We answer the question in the case of Hessenberg determinants of the form

$$H(n) = |HB_n(d_{-1}, 0, 0, \ldots, 0, d_p, 0, \ldots, 0, d_{n-c}, , \ldots, d_{n-1})|$$

with integers p, c.

In the case $c = 0$, we observe that the determinant has to vanish if $n \neq (p+1) \cdot s$ for some multiple s. Otherwise, we can simply expand $H(n) = (-1)^p d_p(n-p) \prod_{l=1}^{p} d_{-1}(n-l)\, HB(n-p-1)$ and get

$$H(n) = \begin{cases} (-1)^{sp} \prod_{l=0}^{s-1} d_p(n-p-l(p+1)) \prod_{k=1}^{p} d_{-1}(n-l(p+1)-k), & \text{if } n = (p+1) \cdot s \\ 0, & \text{otherwise.} \end{cases} \tag{4.12}$$

The case $c > 0$ is more tricky, however. We assume $n \geq 2c+1$ again and observe that

$$H(0) = 1,\ H(1) = \cdots = H(p) = 0 \ \text{ and } \ H(p+1) = d_p(1) \prod_{k=1}^{p} d_{-1}(k).$$

Examining the diagonal generating functions of the upper right corner of the matrix, $d_{n-c}, \ldots, d_{n-1}$, we notice that only $d_{n-1}, d_{n-1-(p+1)}, \ldots, d_{n-1-a(p+1)}$ with $a = \left\lfloor \frac{c}{p+1} \right\rfloor$ contribute to the determinant since the others result in a recursive call of one of $H(1), \ldots, H(p)$. Hence, the recurrence simplifies to

$$\begin{aligned} H(n) &= (-1)^p d_p(n-p) H(n-p) \prod_{l=1}^{p} d_{-1}(n-l) \\ &\quad + \sum_{k=0}^{a} (-1)^{n-1-k(p+1)} d_{n-1-k(p+1)}(k(p+1)+1) \prod_{l=1}^{n-1-k(p+1)} d_{-1}(n-l)\, H(k(p+1)). \end{aligned}$$

Now observe that $H(k(p+1)) = \prod_{l=1}^{k} (-1)^p d_p(l(p+1)) \prod_{j=1}^{p} d_{-1}(l(p+1)-j)$ for $k = 1, \ldots, a$.

Obviously, if $n \neq (p+1) \cdot s$ for some multiple s, all the terms resulting from the first term of the right hand side vanish and we are left with

$$\begin{aligned} H(n) = \sum_{k=0}^{a} \Bigg\{ & (-1)^{n-1-k(p+1)} d_{n-1-k(p+1)}(k(p+1)+1) \\ & \prod_{l=1}^{k} \left[(-1)^p d_p(l(p+1)) \prod_{j=1}^{p} d_{-1}(l(p+1)-j) \right] \prod_{l=1}^{n-1-k(p+1)} d_{-1}(n-l) \Bigg\} \end{aligned} \tag{4.13}$$

Expanding the recurrence for $n = (p+1) \cdot s$, we notice that the last relevant diagonal d_{n-l} vanishes in each level. Multiplying out, we get

$$\begin{aligned} H(n) = & (-1)^{sp} \prod_{l=0}^{s-1} d_p(n-p-l(p+1)) \prod_{l=1}^{p} d_{-1}(n-l(p+1)-k) \\ & + \sum_{j=0}^{a} \Bigg[\prod_{l=1}^{j} \left((-1)^p d_p(n-p-l(p+1)) \prod_{k=1}^{p} d_{-1}(n-l(p+1)-k) \right) \\ & \Bigg\{ \sum_{k=j}^{a} (-1)^{n-1-k(p+1)} d_{n-1-k(p+1)}(k(p+1)+1) \prod_{l=1}^{n-1-k(p+1)} d_{-1}(n-l) \\ & \prod_{l=1}^{k} \left((-1)^p d_p(l(p+1)) \prod_{m=1}^{p} d_{-1}(l(p+1)-m) \right) \Bigg\} \Bigg] \end{aligned} \tag{4.14}$$

If we have constant diagonal generating functions, this simplifies to

$$H(n) = \begin{cases} (-1)^{sp} d_p^s d_{-1}^{sp} + \sum_{k=0}^{a} (k+1)(-1)^{n-1-k} d_{n-1-k(p+1)} d_p^k d_{-1}^{n-1-k}, & \text{if } n = s(p+1) \\ \sum_{k=0}^{a} (k+1)(-1)^{n-1-k} d_{n-1-k(p+1)} d_p^k d_{-1}^{n-1-k}, & \text{otherwise.} \end{cases} \tag{4.15}$$

which completes our investigation of the special case $H(n) = |HB_n(d_{-1}, d_0, 0, \ldots, 0, d_{n-c}, \ldots, d_{n-1})|$ of Hessenberg determinants.

Let us illustrate the last formula in an example.

Example

Consider the Hessenberg determinant of order n generated by the functions $d_{-1} = z, d_1 = b, d_{n-5} = v, d_{n-3} = w, d_{n-2} = x, d_{n-1} = y$:

$$H(n) = \begin{vmatrix} 0 & b & 0 & \ddots & 0 & v & 0 & w & x & y \\ z & 0 & b & 0 & \ddots & 0 & v & 0 & w & x \\ 0 & z & 0 & b & \ddots & \ddots & 0 & v & 0 & w \\ \vdots & \ddots & z & \ddots & \ddots & \ddots & \ddots & 0 & v & 0 \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 & v \\ \vdots & & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & & \ddots & \ddots & 0 & b & 0 & \ddots \\ \vdots & & & & & \ddots & z & 0 & b & 0 \\ \vdots & & & & & & \ddots & z & 0 & b \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & z & 0 \end{vmatrix}.$$

Applying identity (4.15) for $n = 2m \geq 11$, we receive the formula

$$H(n) = (-1)^m (bz)^m + (-1)^{n-1} y z^{n-1} - (-1)^{n-3} 2bwz^{n-2} + (-1)^{n-5} 3b^2 v z^{n-3}.$$

Since n is even, this simplifies to

$$H(n) = (-1)^{\frac{n}{2}} (bz)^{\frac{n}{2}} - yz^{n-1} + 2bwz^{n-2} - 3b^2 vz^{n-3}.$$

4.3 Implementation

We address implementation issues of the Maple package HESSENBERGandCONTINUANT that provides the computation of determinant formulas for specified matrices of the discussed shapes. Examples and further details can be found in the appendix or the on–line help pages.

4.3.1 General Considerations

Determinant order The matrix order can be a positive integer or an expression of the form $n + d$ with integer d and a symbolic variable n.

How do we specify the determinant ? If the determinant order is integer, we can give a piecewise definition of a diagonal generating function like in the previous packages. In the symbolic case, however, we impose the restriction on Hessenberg determinants that only total functions in i are allowed, since the adaption of the formulas for piecewise diagonal function specifications would lead to very complicated coding. Continuants of symbolic order, however, may also be defined in a piecewise fashion. The diagonals of a continuant are specified separately whereas the diagonals of a Hessenberg determinant are specified in a list of two element lists containing the diagonal position (between -1 for the first lower side diagonal, 0 for the main diagonal and up to $n-1$ for the $n-1$st side diagonal) and corresponding generating function.

Each nonzero diagonal has to be specified separately which may seem quite cumbersome regarding dense Hessenberg matrices. In fact it is not possible to specify a Hessenberg determinant of symbolic order with no zero diagonals but since we are only able to derive explicit formulas for certain simple forms of Hessenberg determinants and can only give a recurrence relation for more complicated ones, this restriction makes sense.

Checking the determinant The resulting formula can be checked for specified integer orders (default 4) which is done like in the previous chapters.

4.3.2 The Package functions

The HESSENBERGandCONTINUANT package consists of the following functions:

`Continuant, ContinuantMatrix, HessenbergDet, HessenbergMatrix`.

Continuant The function first checks the input and determines additional informations.

In the case of integer determinant orders it calls the package internal function `HBrec` which recursively computes the specified determinant according to (4.2). The Maple option `remember` is used in this function to avoid multiple identical calls. Each Maple procedure has an associated remember table. The table index is the arguments and the entry is the function value. When a procedure is called, Maple first looks up the remember table and returns the result from there if it already has been computed, otherwise the code of the procedure is executed and the returned result is stored in the remember table under the index of the current arguments. This enables us to keep the relatively simple recursive implementation instead of a more involved iterative one without getting inefficient.

If the determinant order is symbolic we proceed as following:

- If the diagonal generating functions are piecewise defined then the continuant computation is split recursively according to Corollary 14.
- If the main diagonal is zero we compute the determinant formula according to Corollary 12.
- If the main diagonal is sum of the side diagonals in a row then a formula is determined using (4.7).
- If the diagonal generating functions are constant then we return a formula according to (4.4), otherwise the corresponding recurrence relation is returned.

HessenbergDet The function first checks the input and determines additional informations.

In the case of integer determinant orders it calls the package internal function `HBrec` which recursively computes (see above) the specified determinant according to (4.9).

If the determinant order is symbolic we determine the first lower side diagonal function, the main diagonal function, the list of diagonal functions at integer positions `[[p1,f_p1],...,[p_k,f_p_k]]`, and the list of diagonal functions at positions of the form $n-d$ (i.e. those in the upper right corner of the matrix) `[[n-q_c, f_q_c],...,[n-q1, f_q1]]`.

Now we examine the determined lists and try to recognize one of the special cases for which we were able to find formulas:

- If the lower diagonal function is zero or the list of all upper diagonals is empty, we have the case of a triangular determinant and return the product of the main diagonal elements.
- If the main diagonal is zero then it is tested if the list of the nonzero diagonals at integer positions contains only one element, say at position p. In this case we determine a formula according to one of the identities (4.12), (4.13), (4.14), and (4.15). The returned formula is a list containing the result in the case $n = (p+1)k$ and the case $n \neq (p+1)k$.

- If the only nonzero diagonals are the main diagonal, the first lower side diagonal and diagonals at positions $n-d$, we compute the corresponding determinant formula according to Corollary 15.
- If only the first lower side diagonal, main diagonal and the first upper diagonal (at position 1) are nonzero, we have the special case of a continuant and call our function `Continuant`.
- Otherwise, we determine the recurrence relation according to (4.9). If the diagonals are generated by constant functions then it might be possible for some easy cases to obtain a closed form using Maple's `rsolve` function.

ContinuantMatrix This function returns the matrix of the specified continuant if the specification was correct. We use dots "o" to abbreviate the symbolic order. Since Maple treats the dots as usual matrix entries, this function should be used for illustrative purposes only for symbolic orders.

HessenbergMatrix This function returns the specified Hessenberg matrix if the specification was correct. The same remarks as above apply.

4.4 Summary

This chapter focussed on the determinant of two matrix classes defined by diagonal generating functions. First, we examined determinants of tridiagonal matrices following [Met60], the so–called continuants, for which we derived a simple recurrence relation. The origin of this name was explained illustrating the role of polynomials of this kind in the field of continuous fractions. We briefly described the relation to Fibonacci numbers and Euclid's algorithm. It was possible to derive explicit formulas for some special cases of continuants like continuants with a zero main diagonal or continuants where the main diagonal is the sum of the side diagonals of the same row.

Next, we turned to a more general matrix class, the Hessenberg matrices, which we defined using diagonal generating functions. Again, it was possible to establish a recurrence relation for determinants of Hessenberg matrices. However, this recurrence allowed explicit formulas only for very simplified versions of Hessenberg determinants like Hessenberg determinants which had a finite chunk of nonzero diagonals in the upper right corner including the first lower side diagonal and only one other diagonal.

The corresponding Maple package deals with specifications of continuants and Hessenberg determinants (with a restrictive specification facility of Hessenberg determinants of symbolic order) and is able to derive determinant formulas for all the covered general and special cases (see the appendix for examples). Computing integer order determinants using the package functions yields faster results than using the normal det function for higher orders (especially for symbolic entries and in the Hessenberg case).

Chapter 5

Determinants of symmetric matrices

In this chapter we investigate determinant formulas of another important matrix class, the symmetric matrices. We will discuss several kinds of symmetry and the effects on the determinant of such matrices. Two kinds of symmetry will be distinguished, symmetry with respect to a line and symmetry with respect to a point.

5.1 Centrosymmetric Determinants

We will begin with symmetry with respect to a point, the center of a matrix.

Definition 20 *A determinant of order n is called* centrosymmetric *if the reversed rth row yields row $n-r+1$ for all $r = 1, \dots, n$. That is, writing down the rows one after another, the determinant is the same read forward as read backwards.*

Example

C is a centrosymmetric determinant of order 5:

$$C = \begin{vmatrix} a_1 & a_2 & a_3 & a_4 & a_5 \\ b_1 & b_2 & b_3 & b_4 & b_5 \\ c_1 & c_2 & c_3 & c_2 & c_1 \\ b_5 & b_4 & b_3 & b_2 & b_1 \\ a_5 & a_4 & a_3 & a_2 & a_1 \end{vmatrix}.$$

How does this special structure effect the determinant? We will state a result from [Met60] that gives insight into the determinant structure of centrosymmetric matrices and provides an algorithm to compute it efficiently.

We want to compute the centrosymmetric determinant A of order n:

$$A = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1,n-1} & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2,n-1} & a_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{2n} & a_{2,n-1} & \cdots & a_{22} & a_{21} \\ a_{1n} & a_{1,n-1} & \cdots & a_{12} & a_{11} \end{vmatrix}.$$

First, suppose that the order is even and we have $n = 2m$. Performing the elementary row operations $row_l = row_l + row_{2m-l+1}$ and the elementary column operations $col_l = col_l - col_{2m-l+1}$ for $l = 1, \dots, m$, we obtain a square of m^2 zeroes in the upper left corner and the block form

$$\begin{vmatrix} 0 & D \\ D' & * \end{vmatrix}$$

Therefore the determinant A of order n breaks up into two determinants D and D' of order m and we have

$$A = (-1)^m D \cdot D'.$$

The special centrosymmetric structure of A yields that

$$\begin{aligned} D &= |(a_{rs} + a_{rt})|_m \\ &= \begin{vmatrix} a_{1,m} + a_{1,m+1} & a_{1,m-1} + a_{1,m+2} & \cdots & a_{12} + a_{1,2m-1} & a_{11} + a_{1,2m} \\ a_{2,m} + a_{2,m+1} & a_{2,m-1} + a_{2,m+2} & \cdots & a_{22} + a_{2,2m-1} & a_{21} + a_{2,2m} \\ \vdots & \vdots & & \vdots & \vdots \\ a_{m-1,m} + a_{m-1,m+1} & a_{m-1,m-1} + a_{m-1,m+2} & \cdots & a_{m-1,2} + a_{m-1,2m-1} & a_{m-1,1} + a_{m-1,2m} \\ a_{m,m} + a_{m,m+1} & a_{m,m-1} + a_{m,m+2} & \cdots & a_{m,2} + a_{m,2m-1} & a_{m,1} + a_{m,2m} \end{vmatrix} \end{aligned}$$

and

$$\begin{aligned} D' &= |(a_{rt} - a_{rs})|_m \\ &= \begin{vmatrix} a_{m,2m} - a_{m,1} & a_{m,2m-1} - a_{m,2} & \cdots & a_{m,m+2} - a_{m,m-1} & a_{m,m+1} - a_{m,m} \\ a_{m-1,2m} - a_{m-1,1} & a_{m-1,2m-1} - a_{m,2} & \cdots & a_{m-1,m+2} - a_{m-1,m-1} & a_{m-1,m+1} - a_{m-1,m} \\ \vdots & \vdots & & \vdots & \vdots \\ a_{2,2m} - a_{21} & a_{2,2m-1} - a_{22} & \cdots & a_{2,m+2} - a_{2,m-1} & a_{2,m+1} - a_{2,m} \\ a_{1,2m} - a_{11} & a_{1,2m-1} - a_{12} & \cdots & a_{1,m+2} - a_{1,m-1} & a_{1,m+1} - a_{1,m} \end{vmatrix} \end{aligned}$$

for $r, s = 1, \dots, m$ and $t = 2m, 2m-1, \dots, m+1$.

At the moment, every entry of D and D' is a sum of two elements of the original matrix A. We would like to have determinants where every entry consists only one one element of A. Since determinants are multilinear we may break D into a sum of 2^m determinants with "monomial" elements. We observe that for every determinant

$$D_\alpha = D(\alpha_1, \dots, \alpha_m) = \begin{vmatrix} a_{1\alpha_1} & a_{1\alpha_2} & \cdots & a_{1\alpha_m} \\ a_{2\alpha_1} & a_{2\alpha_2} & \cdots & a_{2\alpha_m} \\ \vdots & \vdots & & \vdots \\ a_{m\alpha_1} & a_{m\alpha_2} & \cdots & a_{m\alpha_m} \end{vmatrix}$$

in this sum there is another determinant

$$D_\beta = D(\beta_1, \dots, \beta_m) = \begin{vmatrix} a_{1\beta_1} & a_{1\beta_2} & \cdots & a_{1\beta_m} \\ a_{2\beta_1} & a_{2\beta_2} & \cdots & a_{2\beta_m} \\ \vdots & \vdots & & \vdots \\ a_{m\beta_1} & a_{m\beta_2} & \cdots & a_{m\beta_m} \end{vmatrix},$$

with $\alpha_k + \beta_k = 2m + 1$ for $k = 1, \ldots, m$. Hence we have 2^{m-1} pairs of order m determinants and it is only necessary to compute one determinant of each pair, since the other can be determined from the result.

The signs of D_α and D_β when the columns are arranged in their natural order (i.e. the ascending in the second index of a) are the same if $m(m-1)/2$ is even or opposite if $m(m-1)/2$ is odd. For, if there are g_k numbers following α_k which are smaller than α_k, there are g_k numbers following β_k which are larger than β_k. Therefore g_k is the number of column exchanges due to the correct position of α_k in D_α and $m - k - g_k$ the number of column exchanges due to the correct position of β_k in D_β. This means that the sign factor of D_α is $(-1)^{g_1+\cdots+g_m}$ and the sign factor of D_β is $(-1)^{m(m-1)/2\,-(g_1+\cdots+g_m)}$.

In the case of D' it is obvious that the same 2^m determinants occur as in D. The signs of the various terms will be the same except that when there is an odd number of columns with negative elements, the sign will be changed.

Now we will focus on centrosymmetric determinants of odd orders and present a similar method for computing their determinant.

Assume $n = 2m+1$. Performing elementary row operations $row_l = row_l + row_{2m+2-l}$ and column operations $col_l = col_l - col_{2m+2-l}$ for $l = 1, \ldots, m$, we transform A into a determinant with a rectangle of $m+1$ rows and m columns of zeroes in the upper left-hand corner:

$$\begin{vmatrix} 0 & \bar{D} \\ \bar{D}' & * \end{vmatrix}.$$

Thus, A breaks up into the product of a $m \times m$ determinant $\overline{D}'$and a $(m+1) \times (m+1)$ determinant $\overline{D}$.

We observe that the determinant $\bar{D}$ in the lower left–hand corner is identical to D' and the $m \times m$ subdeterminant of $\overline{D}$ in the upper right–hand corner is identical to D, thus only the first column and the last row of $\overline{D}$ disturb. Expanding the last row of $\overline{D}$ gives the formula

$$A = -\bar{D}' \cdot \bar{D},$$

with

$$\bar{D} = a_{m+1,m+1} D + \sum_{k=1}^{m} (-1)^k a_{m+1,m+1+k} \bar{D}_{m+1,k+1}.$$

It is possible to compute $\bar{D}_{m+1,k+1}$ using the same methods as above.

Unfortunately, the described methods do not enable us to obtain explicit formulas for centrosymmetric determinants of symbolic order n but it clarifies its structure. Moreover, in this case, we reduced a determinant problem of order n to two determinant problems of order $\lfloor \frac{n}{2} \rfloor$ and $\lceil \frac{n}{2} \rceil$ of special structure and thus the presented methods are more efficient than general computation methods for determinants with symbolic entries.

Similar methods for the case of skew–centrosymmetric determinants can be found in [Met60].

After discussing the general problem of centrosymmetric determinants, we will turn to a few special cases that allow us to derive explicit formulas.

We will derive formulas for a more general non–centrosymmetric case which obviously also apply for the special centrosymmetric case.

Consider the determinant $|A|$ of order n where row i is generated by the constant function b_i, apart from the main diagonal which is generated by a_i:

$$|A| = \begin{vmatrix} a_1 & b_1 & \cdots & \cdots & b_1 \\ b_2 & a_2 & b_2 & \cdots & b_2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ b_{n-1} & \cdots & b_{n-1} & a_{n-1} & b_{n-1} \\ b_n & \cdots & \cdots & b_n & a_n \end{vmatrix}.$$

If we successively subtract the first column from all the other columns then $|A|$ is transformed into arrow form:

$$|A| = \begin{vmatrix} a_1 & b_1 - a_1 & b_1 - a_1 & \cdots & b_1 - a_1 \\ b_2 & a_2 - b_2 & 0 & \cdots & 0 \\ b_3 & 0 & a_3 - b_3 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ b_n & 0 & \cdots & 0 & a_n - b_n \end{vmatrix}.$$

Using the formula from the second chapter, this yields

$$|A| = a_1 \prod_{l=2}^{n} (a_l - b_l) + \prod_{l=1}^{n} (a_l - b_l) \sum_{k=2}^{n} \frac{b_k}{a_k - b_k}.$$

which can be simplified to

$$|A| = \prod_{l=1}^{n} (a_l - b_l) \left[1 + \sum_{k=1}^{n} \frac{b_k}{a_k - b_k} \right]. \tag{5.1}$$

For centrosymmetric determinants of this kind, we have $a_i = a_{n-i+1}$ and $b_i = b_{n-i+1}$ for $i = 1, \ldots, \lfloor \frac{n}{2} \rfloor$, thus (5.1) simplifies to

$$|A| = \prod_{l=1}^{\lfloor \frac{n}{2} \rfloor} (a_l - b_l)^2 \left[1 + 2 \sum_{k=1}^{\lfloor \frac{n}{2} \rfloor} \frac{b_k}{a_k - b_k} \right]. \tag{5.2}$$

Let us discuss another determinant problem. Consider the determinant of a matrix M which is defined by

$$m_{ij} = \begin{cases} a_i & , i = j, \\ b_i & , i < j, \\ b_{i-1} & , i > j. \end{cases}$$

Hence, we are interested in the following determinant.

$$|M| = \begin{vmatrix} a_1 & b_1 & b_1 & \cdots & b_1 \\ b_1 & a_2 & b_2 & \cdots & b_2 \\ b_2 & b_2 & a_3 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & b_{n-1} \\ b_{n-1} & b_{n-1} & \cdots & b_{n-1} & a_n \end{vmatrix}.$$

Performing the elementary column operations $col_k = col_k - col_1$ for $k = 2, \ldots, n$ we may transform $|M|$ to

$$|M| = \begin{vmatrix} a_1 & b_1 - a_1 & b_1 - a_1 & b_1 - a_1 & \cdots & b_1 - a_1 \\ b_1 & a_2 - b_1 & b_2 - b_1 & b_2 - b_1 & \cdots & b_2 - b_1 \\ b_2 & 0 & a_3 - b_2 & b_3 - b_2 & \ddots & \vdots \\ \vdots & \vdots & 0 & \ddots & \ddots & b_{n-2} - b_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \ddots & b_{n-1} - b_{n-2} \\ b_{n-1} & 0 & 0 & \cdots & 0 & a_n - b_{n-1} \end{vmatrix}.$$

Now we successively subtract col_k from col_{k+1} for $k = n-1, n-2, \ldots, 2$ and get

$$|M| = \begin{vmatrix} a_1 & b_1 - a_1 & 0 & \cdots & 0 \\ b_1 & a_2 - b_1 & b_2 - a_2 & \ddots & \vdots \\ b_2 & 0 & a_3 - b_2 & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & b_{n-1} - a_{n-1} \\ b_{n-1} & 0 & \cdots & 0 & a_n - b_{n-1} \end{vmatrix},$$

which is a 4–oriented 7–form determinant. Transposing, we can apply the theorem for standard 7–forms and obtain the formula

$$\begin{aligned} |A| &= (a_1 - b_1) \textstyle\sum_{l=2}^{n} (-1)^{n+l} b_{l-1} \prod_{k=2}^{l-1} (a_k - b_k) \prod_{k=l+1}^{n} (b_{k-1} - a_k) \\ &\quad + (-1)^{n+1} a_1 \textstyle\prod_{k=2}^{n} (b_{k-1} - a_k), \end{aligned}$$

which may be simplified to

$$|A| = \sum_{l=2}^{n} b_{l-1} \prod_{k=1}^{l-1} (a_k - b_k) \prod_{k=l+1}^{n} (a_k - b_{k-1}) + a_1 \prod_{k=2}^{n} (a_k - b_{k-1}). \tag{5.3}$$

We omit the simplification in the centrosymmetric case since it does not simplify the structure of the formula.

5.2 Axisymmetric Determinants

The best known type of symmetry is the axisymmetry with respect to the main diagonal.

Definition 21 *A matrix A is called* (axi)symmetric *if $a_{ij} = a_{ji}$ for all $i, j = 1, \ldots, n$.*

Axisymmetric matrices arise in many applications, of special interest are the positive definite symmetric matrices for which there are special algorithms, e.g. Cholesky factorization. Being interested in general determinant formulas we will have to restrict the class of symmetric determinants a little since axisymmetry is too general to obtain determinant formulas for the general case.

Following, we will show how a special class of axisymmetric determinants can be reduced to arrow form or tridiagonal form respectively.

Consider axisymmetric matrices A that are generated by two functions f and g, such that $a_{ii} = f(i) =: f_i$ and $a_{ij} = g(i) =: g_i$ for $i < j$. Hence we are interested in a determinant formula of matrices of the following type:

$$A = \begin{pmatrix} f_1 & g_1 & g_1 & \cdots & g_1 \\ g_1 & f_2 & g_2 & \ddots & \vdots \\ g_1 & g_2 & \ddots & \ddots & g_{n-2} \\ \vdots & \ddots & \ddots & f_{n-1} & g_{n-1} \\ g_1 & \cdots & g_{n-2} & g_{n-1} & f_n \end{pmatrix}.$$

Performing the elementary column operations $col_k = col_k - col_{k-1}$ and afterwards $row_k = row_k - row_{k-1}$ for $k = n, n-1, \ldots, 2$, we may transform $|A|$ into continuant form:

$$|A| = \begin{vmatrix} f_1 & -f_1+g_1 & 0 & \cdots & 0 \\ -f_1+g_1 & f_1-2g_1+f_2 & -f_2+g_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -f_{n-2}+g_{n-2} & f_{n-2}-2g_{n-2}+f_{n-1} & -f_{n-1}+g_{n-1} \\ 0 & \cdots & 0 & -f_{n-1}+g_{n-1} & f_{n-1}-2g_{n-1}+f_n \end{vmatrix}.$$

This looks suggestive that we might be able to apply identity (4.7) getting a reasonably nice formula but unfortunately the assumptions are a little different, so we have to settle with the recurrence

$$|A| = K(n) = (f_{n-1} + 2g_{n-1} + f_n) \cdot K(n-1) - (f_{n-1} - g_{n-1})^2 \cdot K(n-2)$$

with $K(1) = f_1$ and $K(0) = 1$. Having generating functions f and g with $a_{ii} = f_i$ and $a_{ij} = g_j$ for $i < j$ yields a similar recurrence.

Let us pick out two easy special cases:

If $g_i = f_i$ for $i = 1, \ldots, n-1$ then we obtained diagonal form after the transformation and have the determinant formula

$$|A| = f_1 \prod_{l=2}^{n} (f_l - f_{l-1}). \tag{5.4}$$

If g is constant we subtract the first row from all other rows instead and reach arrow form

$$|A| = \begin{vmatrix} f_1 & g & g & \cdots & g \\ g-f_1 & f_2-g & 0 & \cdots & 0 \\ g-f_1 & 0 & f_3-g & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ g-f_1 & 0 & \cdots & 0 & f_n-g \end{vmatrix}.$$

Now we may use the result on arrow determinants of the second chapter to obtain the formula

$$|A| = \prod_{l=1}^{n}(f_l - g)\,(1 + \sum_{k=1}^{n} \frac{g}{f_k - g}). \tag{5.5}$$

Let us focus on another special axisymmetric matrix class. Consider matrices generated by functions m and p, where $a_{ij} = p(i)p(j)$ for $i < j$ and $a_{ii} = m(i)$:

$$A = \begin{pmatrix} m(1) & p(1)p(2) & \cdots & p(1)p(n) \\ p(1)p(2) & m(2) & \ddots & \vdots \\ \vdots & \ddots & \ddots & p(n-1)p(n) \\ p(1)p(n) & \cdots & p(n-1)q(n) & m(n) \end{pmatrix}$$

Performing the column operations $col_k = col_k - \frac{p(k)}{p(1)} \cdot col_1$ for $k = 2, \dots, n$ (provided $p(1) \neq 0$) we obtain arrow form for the determinant:

$$|A| = \begin{vmatrix} m(1) & \frac{p(2)}{p(1)}(p(1)^2 - m(1)) & \frac{p(3)}{p(1)}(p(1)^2 - m(1)) & \cdots & \cdots & \frac{p(n)}{p(1)}(p(1)^2 - m(1)) \\ p(1)p(2) & m(2) - p(2)^2 & 0 & \cdots & \cdots & 0 \\ p(1)p(3) & 0 & m(3) - p(3)^2 & 0 & \cdots & 0 \\ \vdots & \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ p(1)p(n) & 0 & 0 & \cdots & 0 & m(n) - p(n)^2 \end{vmatrix}.$$

Extracting the factor $(p(1)^2 - m(1))/p(1)$ from the first row and the factor $p(1)$ from first column (provided $p(1)^2 - m(1) \neq 0$), this simplifies to

$$|A| = (p(1)^2 - m(1)) \cdot \begin{vmatrix} \frac{m(1)}{p(1)^2 - m(1)} & p(2) & p(3) & \cdots & p(n) \\ p(2) & m(2) - p(2)^2 & 0 & \cdots & 0 \\ p(3) & 0 & m(3) - p(3)^2 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ p(n) & 0 & \cdots & 0 & m(n) - p(n)^2 \end{vmatrix}.$$

We obtained a simple arrow form that we can transform into triangular form via

$$row_1 = row_1 - \frac{p(k)}{m(k) - p(k)^2} \cdot row_k \text{ for } k = 2, \dots, n.$$

Hence we obtain the following determinant formula

$$|A| = m(1) \prod_{k=2}^{n} (m(k) - p(k)^2) \; + \sum_{l=2}^{n} p(l)^2 \prod_{\substack{k=1 \\ k \neq l}}^{n} (m(k) - p(k)^2),$$

which may be further simplified to

$$|A| = \prod_{k=1}^{n}(m(k) - p(k)^2) \cdot \left[1 + \sum_{k=1}^{n} \frac{p(k)^2}{m(k) - p(k)^2}\right]. \tag{5.6}$$

Let us illustrate this formula with a little example.

Example

Consider the axisymmetric determinant

$$|A| = \begin{vmatrix} x_1^2+1 & x_1x_2 & x_1x_3 & \cdots & x_1x_n \\ x_1x_2 & x_2^2+1 & x_2x_3 & \cdots & x_2x_n \\ x_1x_3 & x_2x_3 & x_3^2+1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & x_{n-1}x_n \\ x_1x_n & x_2x_n & \cdots & x_{n-1}x_n & x_n^2+1 \end{vmatrix}$$

with main diagonal generating function $m(i) = x_i^2 + 1$ and function x_ix_j generating a_{ij} for $i < j$ (that is $p(i) = x_i$ and $p(j) = x_j$). Applying identity (5.6) yields the nice determinant formula

$$|A| = 1 + x_1^2 + x_2^2 + \cdots + x_n^2.$$

In closing, we discuss another special class of axisymmetric determinants. Let us consider determinants $|A|$ generated by the function $a_{ij} = c \cdot (j - i) + k$:

$$|A| = \begin{vmatrix} k & c+k & 2c+k & \cdots & (n-1)c+k \\ c+k & k & c+k & \ddots & \vdots \\ 2c+k & c+k & k & \ddots & 2c+k \\ \vdots & \ddots & \ddots & \ddots & c+k \\ (n-1)c+k & \cdots & 2c+k & c+k & k \end{vmatrix}.$$

We can simplify this significantly by performing the operations $row_k = row_k - row_{k+1}$ for $k = 1, \ldots, n-1$, and get

$$|A| = \begin{vmatrix} -c & c & c & \cdots & c \\ -c & -c & c & \cdots & c \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ -c & \cdots & -c & -c & c \\ (n-1)c+k & \cdots & 2c+k & c+k & k \end{vmatrix}.$$

It remains to add the first column to all the other columns to obtain a triangular determinant

$$|A| = \begin{vmatrix} -c & 0 & 0 & \cdots & 0 \\ -c & -2c & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ -c & -2c & \cdots & -2c & 0 \\ (n-1)c+k & (2n-3)c+2k & & nc+2k & (n-1)c+2k \end{vmatrix}.$$

Hence we receive the formula

$$|A| = (-1)^{n-1}c(2c)^{n-2}((n-1)c + 2k). \tag{5.7}$$

Similar proceeding, using column operations, establishes that the determinant vanishes if $a_{ij} = c \cdot (i + j) + d$.

Example

Consider the axisymmetric determinant $|A|$ of order n generated by $a_{ij} = j - i + 1$:

$$|A| = \begin{vmatrix} 1 & 2 & 3 & \cdots & n \\ 2 & 1 & \ddots & \ddots & \vdots \\ 3 & \ddots & \ddots & \ddots & 3 \\ \vdots & \ddots & \ddots & 1 & 2 \\ n & \cdots & 3 & 2 & 1 \end{vmatrix}.$$

Using (5.7), we may establish the following identity

$$|A| = (-1)^{n-1}(n+1) \cdot 2^{n-2}.$$

While we cannot derive a formula for the case $a_{ij} = c \cdot (j - i) + d$ with a maindiagonal different from d, it can be shown that the case $a_{ij} = c \cdot (i + j) + d$ with a nonzero main diagonal can be reduced to a oriented "fat" R–form determinant: Consider the determinant of the axisymmetric matrix A of dimension n with $a_{ii} = k_i$ and $a_{ij} = c \cdot (i + j) + d$ otherwise. Performing the row operations as above, we get

$$\begin{vmatrix} -3c - d + k_1 & 3c + d - k_2 & -c & \cdots & -c \\ -c & -5c - d + k_2 & 5c + d - k_3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -c \\ -c & \cdots & -c & -(2n-1)c - d + k_{n-1} & (2n-1)c + d - k_n \\ (n+1)c + d & \cdots & (2n-2)c + d & (2n-1)c + d & k_n \end{vmatrix}.$$

Subtracting the first column from all the other columns, we are left with a oriented fat R–form determinant

$$\begin{vmatrix} -3c - d + k_1 & 6c + 2d - k_1 - k_2 & 2c + d - k_1 & 2c + d - k_1 & \cdots & 2c + d - k_1 \\ -c & -4c - d + k_2 & 6c + d - k_3 & 0 & \cdots & 0 \\ -c & 0 & -6c - d + k_3 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 2(n-1)c + d - k_{n-1} & 0 \\ -c & 0 & \cdots & 0 & -2(n-1)c - d + k_{n-1} & 2nc + d - k_n \\ (n+1)c + d & c & \cdots & (n-3)c & (n-2)c & -(n+1)c - d + k_n \end{vmatrix}.$$

We omit the resulting involved formula that is even too complicated to be handled by Maple using the corresponding implemented Maple package function.

In the following section we will discuss how to obtain formulas for skewsymmetric determinants of this type.

5.3 Zero–axial skew determinants

Now we take a look at a special case of axisymmetric determinants and show its connection to matching theory.

Definition 22 *A matrix A is called* zero–axial skew matrix *if $a_{ij} = -a_{ji}$ and $a_{ii} = 0$. Its corresponding determinant is called* zero-axial skew determinant.

Example

$$A = \begin{pmatrix} 0 & a_{12} & a_{13} & a_{14} \\ -a_{21} & 0 & a_{23} & a_{24} \\ -a_{31} & -a_{32} & 0 & a_{34} \\ -a_{41} & -a_{42} & -a_{43} & 0 \end{pmatrix}.$$

A is a zero-axial skew matrix.

Let us begin with a few simple observations concerning zero–axial skew determinants.

Observation

A zero–axial skew determinant of odd order vanishes.

If we change the signs of all elements, e.g. by multiplying each column with -1, the determinant of a zero–axial skew matrix does not change since transposing the matrix would yield the same. However, a determinant of odd order is changed in sign and therefore has to be zero.

What happens for even orders? We present an old result taken from [Met60].

Theorem 23 *A zero–axial skew determinant of even order is the square of a rational function in its elements.*

Proof.

For any zero–axial skew determinant $|A|$ of order n we have

$$|A_{rr}| \cdot |A_{ss}| - |A_{rs}| \cdot |A_{sr}| = |A| \cdot |A_{rs,rs}.| \tag{5.8}$$

according to [Met60] (page 393).

If n is even then $|A_{rr}| = |A_{ss}| = 0$ for all r, s because of the last observation and the fact that all coaxial minors of a zero–axial skew determinant are zero-axial skew again. Moreover, $|A_{rs}| = -|A_{sr}|$. Therefore (5.8) simplifies to

$$|A_{rs}|^2 = |A| \cdot |A_{rs,rs}| \text{ or } |A| = \frac{|A_{rs}|^2}{|A_{rs,rs}|}. \tag{5.9}$$

We may use (5.9) for an inductive argument to prove the theorem, for it shows that the theorem is true for $|A|$ of even order n if it is true for $|A_{rs,rs}|$ of even order $n-2$. Since the theorem is obviously true for $n = 2$, it must also hold in general.

$\square$

Looking at the coaxial minors of order two of $|A|_{2n}$ we see that their product $a_{12}^2 a_{34}^2 \cdots a_{2n-1,2n}^2$ is a term of the determinant and that therefore one square root of $|A|$ contains the term $+a_{12}a_{34}\cdots a_{2n-1,2n}$ and the other $-a_{12}a_{34}\cdots a_{2n-1,2n}$. This leads us to the following important definition.

Definition 23 *The square root of a zero–axial skew determinant $|A| = |(a_{ij})|$ of order $2n$ which contains as a positive term $a_{12}a_{34}\cdots a_{2n-1,2n}$ is called a* Pfaffian *function of the elements lying on the upper side of the zero main diagonal and will be denoted as Pf(A).*

Hence we can restate the previous theorem as

$$|A| = \mathrm{Pf}(A)^2 \tag{5.10}$$

for zero–axial skew matrices A of even order.

Pfaffian functions can be defined in many ways, we will use another (see [Knu96]) to illustrate the connection of Pfaffians to perfect matchings:

For each possible partition $P = \{\{i_1, j_1\}, \ldots, \{i_n, j_n\}\}$ of the set $\{1, \ldots, 2n\}$ into ordered pairs, form the expression

$$a_P = \mathrm{sgn}\begin{pmatrix} 1 & 2 & \cdots & 2n-1 & 2n \\ i_1 & j_1 & \cdots & i_n & j_n \end{pmatrix} a_{i_1 j_1}\cdots a_{i_n j_n},$$

where

$$\begin{pmatrix} 1 & 2 & \cdots & 2n-1 & 2n \\ i_1 & j_1 & \cdots & i_n & j_n \end{pmatrix}$$

is a permutation of the elements $1, 2, \ldots, 2n-1, 2n$ and sgn denotes the sign of this permutation. An alternative definition of the Pfaffian of a matrix A is

$$\mathrm{Pf}(A) = \sum_P a_P. \tag{5.11}$$

We now state some fundamental results following [LP86] relating the Pfaffian to perfect matchings in a graph.

Recall that a *matching* in a graph $G = (V, E)$ is a set of edges that have no vertices in common. A matching is *perfect* if it covers all of $V(G)$. A simple theorem tells us that a bipartite graph G has a perfect matching if and only if the determinant $|M(x)|$ is not identically zero for $M(x) = (m_{ij})$ and

$$m_{ij} = \begin{cases} x_e, & \text{if } e = (v_i, v_j), \\ 0, & \text{if } v_i, v_j \text{ non-adjacent.} \end{cases}$$

.

Surprisingly, the use of determinants can be extended to non-bipartite graphs. Consider a graph G with $V(G) = \{v_1, \ldots, v_k\}$. To each edge e we assign a variable x_e and define the matrix $A(x)$ as follows:

$$A(x) = (a_{ij})_{k\times k}$$

where

$$a_{ij} = \begin{cases} x_e, & \text{if } e = (v_i, v_j), \\ -x_e, & \text{if } e = (v_j, v_i), \\ 0, & \text{if } v_i \text{ and } v_j \text{ non-adjacent.} \end{cases}$$

The following result is due to Tutte (1947).

Theorem 24 *Let G be any graph and let $A(x)$ be defined as above. Then graph G has a perfect matching if and only if the determinant $|A(x)|$ is not identically zero.*

Proof.

If k is odd then G has no matching, of course, and moreover, $|A| = 0$ since the determinant of a zero–axial skew matrix of odd dimension vanishes. Thus the theorem is true for graphs with an odd number of nodes. So assume that $k = |V(G)| = 2n$.

Now each nonzero term a_P in the defining identity (5.11) of $\mathrm{Pf}(A(x))$ corresponds to a perfect matching of G and vice versa. Furthermore, different perfect matchings of G correspond to terms consisting of different variables. Hence $\mathrm{Pf}(A(x)) \equiv 0$ if and only if G has no perfect matching. Using the result (5.10), the theorem follows.

□

Finally, let us point out that the existence criterion given in the theorem can be used to obtain a probabilistic algorithm for finding a perfect matching in a graph and that it is even possible to use Pfaffians to count the number of different perfect matchings (for details see [LP86]) . Other applications of Pfaffians include plane partitions and problems in differential geometry.

After this short excursion, let us turn back to zero–axial skew determinants. A more constructive definition of the Pfaffian of a zero–axial skew matrix A of order n would be the recursion:

$$\mathrm{Pf}_n(A) = \sum_{k=2}^{n} (-1)^k a_{1k} \cdot \mathrm{Pf}_{n-2}(|A_{11,kk}|) \tag{5.12}$$

for even n and the base case $\mathrm{Pf}_0() = 1$ which can be viewed as "minor expansion" for Pfaffians. See [DW95] for a proof.

It is interesting to note that Pfaffians are more fundamental than matrices:

Theorem 25 *Any determinant may be expressed as a Pfaffian of the same order.*

Proof.

Consider the the determinant Δ of order $2n$:

$$\Delta = \begin{vmatrix} \frac{1}{2}(a_{11}-a_{11}) & \cdots & \frac{1}{2}(a_{1n}+a_{n1}) & \frac{1}{2}(a_{1n}+a_{n1}) & \cdots & \frac{1}{2}(a_{11}+a_{11}) \\ \vdots & & \vdots & \vdots & & \vdots \\ \frac{1}{2}(a_{n1}-a_{1n}) & \cdots & \frac{1}{2}(a_{nn}-a_{nn}) & \frac{1}{2}(a_{nn}+a_{nn}) & \cdots & \frac{1}{2}(a_{n1}+a_{1n}) \\ -\frac{1}{2}(a_{n1}+a_{1n}) & \cdots & -\frac{1}{2}(a_{nn}+a_{nn}) & -\frac{1}{2}(a_{nn}-a_{nn}) & \cdots & -\frac{1}{2}(a_{n1}-a_{1n}) \\ \vdots & & \vdots & \vdots & & \vdots \\ -\frac{1}{2}(a_{11}+a_{11}) & \cdots & -\frac{1}{2}(a_{1n}+a_{n1}) & -\frac{1}{2}(a_{1n}-a_{n1}) & \cdots & -\frac{1}{2}(a_{11}-a_{11}) \end{vmatrix},$$

which is zero–axial and skew centrosymmetric. Performing the elementary operations $row_k = row_k + row_{n-k+1}$ and $col_{n-k} = col_{n-k} + col_l$ for $k = 1, \ldots, n$, we may reduce Δ to the product of two $n \times n$ determinants. A little thought establishes that we now have $\Delta = |A|^2$ with $A = (a_{ij})_{n\times n}$. Hence $|A| = \Delta^{1/2} = \mathrm{Pf}(\Delta)$. Since $|A|$ is an arbitrary determinant, the theorem follows.

□

So far, we examined only zero–axial skew determinants. Sometimes, skew (axi)symmetric determinants are defined to have a possibly nonzero main diagonal. What happens to general skew determinants, i.e. determinants that differ from the zero–axial skew determinants only in a nonzero main diagonal ? It is interesting to see that we may express a skew determinant in terms of the main diagonal elements and zero–axial skew determinants.

Theorem 26 *Let $|M|$ be a skew determinant of order n,*

$$|M| = \begin{vmatrix} x_1 & a_{12} & \cdots & a_{1n} \\ -a_{12} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n-1,n} \\ -a_{1n} & \cdots & -a_{n-1,n} & x_n \end{vmatrix},$$

and $|A|$ the zero–axial skew determinant obtained from $|M|$,

$$|A| = \begin{vmatrix} 0 & a_{12} & \cdots & a_{1n} \\ -a_{12} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n-1,n} \\ -a_{1n} & \cdots & -a_{n-1,n} & 0 \end{vmatrix}.$$

If n is even, we have

$$|M| = |A| + \sum_{1\le i_1<i_2\le n} x_{i_1}x_{i_2}|A_{i_1i_2,i_1i_2}| + \sum_{1\le i_1<i_2<i_3<i_4\le n} x_{i_1}x_{i_2}x_{i_3}x_{i_4}|A_{i_1i_2i_3i_4,i_1,i_2,i_3,i_4}| + \cdots + \sum x_{i_1}\cdots x_{i_{n-2}}|A_{i_1\cdots i_{n-2},i_1\cdots i_{n-2}}| + x_1\cdots x_n.$$

If n is odd, we have

$$|M| = \sum_{1\le i_1\le n} x_{i_1}|A_{i_1,i_1}| + \sum_{1\le i_1<i_2<i_3\le n} x_{i_1}x_{i_2}x_{i_3}|A_{i_1i_2i_3,i_1i_2i_3}| + \cdots + \sum_{1\le i_1\cdots i_{n-2}\le n} x_{i_1}\cdots x_{i_{n-2}}|A_{i_1\cdots i_{n-2},i_1\cdots i_{n-2}}| + x_1\cdots x_n.$$

Proof.

We expand the skew determinant by Cayley's theorem (see pages 107ff. in [Met60]) and observe that the terms having an odd number of diagonal elements vanish.

□

Note that, if $x_1 = \cdots = x_n = x$, we have a power series in x with the squares of Pfaffians as coefficients.

This completes our presentation of general zero–axial skew determinants. Further results can be found in [Met60] and a brief history of Pfaffians is contained in a paper by Knuth [Knu96].

Now we want to turn to special cases of skewsymmetric determinants. Recall the special matrix classes of the previous section. We will try to obtain similar results for the skewsymmetric case.

Consider a skewsymmetric matrix of the form $a_{ij} = g = -a_{ji}$ with maindiagonal $a_{ii} = f_i$:

$$|A| = \begin{vmatrix} f_i & g & \cdots & g \\ -g & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & g \\ -g & \cdots & -g & f_n \end{vmatrix}.$$

Performing the row operations $row_k = row_k - row_{k-1}$ for $k = n, \ldots, 2$, this reduces to a 7–form determinant:

$$|A| = \begin{vmatrix} f_1 & g & g & \cdots & g \\ -f_1 - g & f_2 - g & 0 & \cdots & 0 \\ 0 & -f_2 - g & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & f_{n-1} - g & 0 \\ 0 & \cdots & 0 & -f_{n-1} - g & f_n - g \end{vmatrix},$$

whose formula can be obtained using results from the second chapter. Since the resulting formula is somewhat involved, we omit it and point out the possibility to derive it with the corresponding Maple package function `Form7`.

Recurrence relations similar to the previous section for the case $a_{ii} = f_i$ and $a_{ij} = g_i = -a_{ji}$ or $a_{ij} = g_j = -a_{ji}$ can be obtained using the same techniques.

In closing, we consider a skewsymmetric matrix of the form $a_{ij} = c \cdot (i - j) + d = -a_{ji}$. Applying the same technique of the previous section, subtracting subsequent rows $row_k = row_k - row_{k+1}$, we obtain

$$|A| = \begin{vmatrix} 2d - c & -c & \cdots & -c & -c \\ -c & 2d - c & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & -c & -c \\ -c & \cdots & -c & 2d - c & -c \\ (n-1)c - d & \cdots & 2c - d & c - d & d \end{vmatrix},$$

which simplifies to 3–oriented arrow form after adding the last column to all the other columns:

$$|A| = \begin{vmatrix} 2d & 0 & \cdots & 0 & -c \\ 0 & 2d & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & -c \\ 0 & \cdots & 0 & 2d & -c \\ (n-1)c - 2d & \cdots & 2c - 2d & c - 2d & d \end{vmatrix}.$$

Using the arrow form formula from the second chapter, we receive

$$|A| = (2d)^{n-1} \left[d - (n-1)c + \sum_{l=1}^{n-1} \frac{c^2}{2d}(n-l) \right]. \tag{5.13}$$

Let us turn to the case of $a_{ij} = c \cdot (i + j) + d = -a_{ji}$. After performing our standard trick, we get

$$|A| = \begin{vmatrix} 5c + 2d & -c & -c & \cdots & -c \\ c & 9c + 2d & -c & \cdots & -c \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ c & \cdots & c & (4(n-1)+1)c + 2d & -c \\ -(n+1)c - d & \cdots & -(2(n-1)-1)c - d & -2(n-1)c - d & 2nc + d \end{vmatrix},$$

repeating the row operations $row_k = row_k - row_{k+1}$ for $k = 1, \ldots, n-2$, we are left with

$$\begin{vmatrix} 4c+2d & -10c-2d & 0 & 0 & \cdots & 0 \\ 0 & 8c+2d & -14c-2d & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 4(n-2)c+2d & -(4(n-1)+2)c-2d & 0 \\ c & \cdots & c & c & (4(n-1)+1)c+2d & -c \\ -(n+1)c-d & \cdots & -(2n-3)c-d & -(2n-2)c-d & -(2n-1)c-d & 2nc+d \end{vmatrix},$$

that can be transformed into a fat N–form determinant (swapping the $n-1$st row up to the top) whose formula can be obtained using the results of the second chapter. Since the resulting formula looks rather nasty, we omit it and point out the possibility to obtain it using the corresponding Maple package function `Nform`.

It remains to note that having a different maindiagonal prevents us from successfully applying these techniques.

5.4 Persymmetric Determinants

Let us have a look at yet another type of symmetry.

Definition 24 *A $n \times n$ matrix A is called* persymmetric *if it is symmetric with respect to its northeast southwest diagonal, i.e. $a_{ij} = a_{n-j+1,n-i+1}$ for all i and j.*

This is equivalent to requiring $A = EA^TE$ with

$$E = \begin{vmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \cdot & \cdot & 0 \\ 0 & \cdot & \cdot & \cdots \\ 1 & 0 & \cdots & 0 \end{vmatrix}.$$

The determinant $|A|$ of a persymmetric matrix is called persymmetric determinant.

Note that persymmetry differs from the previously discussed axisymmetry only in that we now have symmetry with respect to the counter main diagonal instead of the main diagonal.

There are several matrix classes that belong to the class of persymmetric matrices. We will discuss two of them in the following: The Toeplitz matrices and the Circulants.

5.4.1 Toeplitz Matrices

Matrices whose entries are constant along each diagonal arise in many applications (e.g. differential equations) and are called Toeplitz matrices. We give a formal definition.

Definition 25 *A $n\times n$ matrix A is called* Toeplitz matrix *if there are constant diagonal generating functions*

$$d_{-(n-1)}, \ldots, d_0, \ldots, d_{n-1}$$

such that $a_{ij} = d_{j-i}$ for all i and j.

Obviously, Toeplitz matrices are persymmetric. Toeplitz matrices and Toeplitz systems have been investigated rather deeply . There is a variety of fast algorithms to solve Toeplitz systems, the special structure of Toeplitz matrices enables us to obtain a running time of $O(n^2)$ to solve linear Toeplitz systems (see [GvL96]).

Since we are interested in general determinant formulas it is necessary to restrict us to special forms of Toeplitz matrices.

At first, we will investigate a matrix class that consists of only two distinct elements. All diagonals are generated using either one of them.

Definition 26 *A matrix M of order n is called a* two element diagonal matrix *if all of its diagonals are generated by two distinct elements a and b. If the "a" diagonals are at position $p_1, p_2, \dots, p_k$ and the "b" diagonals at the remaining positions, we will write $M_n(a, b, [p_1, p_2, \dots, p_k])$.*

Example

$$M_6(a,b,[-1,0,2]) = \begin{pmatrix} a & b & a & b & b & b \\ a & a & b & a & b & b \\ b & a & a & b & a & b \\ b & b & a & a & b & a \\ b & b & b & a & a & b \\ b & b & b & b & a & a \end{pmatrix}.$$

How does the determinant of two element diagonal matrices look like? Since the matrix is dense, minor expansion does not look like a good idea, especially, since we only have two distinct elements. Elementary row and column operations seem to be more in place here.

First, we will look at the special case that we have a a band without holes between the main diagonal and a diagonal at position $k \geq 0$. We will proceed as follows:

Starting with

$$|M_n(a,b,[0,1,\dots,k])| = \begin{vmatrix} a & \cdots & a & b & \cdots & b \\ b & \ddots & & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & \ddots & b \\ \vdots & & \ddots & \ddots & & a \\ \vdots & & & \ddots & \ddots & \vdots \\ b & \cdots & \cdots & \cdots & b & a \end{vmatrix}$$

1. Subtract column n from column $n-l$ for $l = 1, \dots, n-1$ and get

$$\begin{vmatrix} a-b & a-b & \cdots & a-b & a-b & 0 & \cdots & 0 & b \\ 0 & \ddots & & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & 0 & b \\ 0 & \cdots & 0 & a-b & a-b & \cdots & a-b & a-b & b \\ -a+b & \cdots & -a+b & -a+b & 0 & 0 & \cdots & 0 & a \\ -a+b & \cdots & -a+b & -a+b & -a+b & 0 & \cdots & 0 & a \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ -a+b & \cdots & -a+b & -a+b & -a+b & \cdots & -a+b & 0 & a \\ -a+b & \cdots & -a+b & -a+b & -a+b & \cdots & -a+b & -a+b & a \end{vmatrix} \tag{5.14}$$

2. We now subtract row n from row $n-l$ for $l=1,\dots,k-1$ and get

$$\begin{vmatrix}
a-b & a-b & \cdots & a-b & 0 & \cdots & 0 & b \\
0 & a-b & a-b & \cdots & a-b & \ddots & \vdots & b \\
\vdots & \ddots & \ddots & \ddots & & \ddots & 0 & \vdots \\
0 & \cdots & 0 & a-b & a-b & \cdots & a-b & b \\
0 & \cdots & 0 & 0 & a-b & \cdots & a-b & 0 \\
\vdots & & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & a-b & 0 \\
-a+b & \cdots & -a+b & -a+b & \cdots & -a+b & -a+b & a
\end{vmatrix}.$$

3. Now, we are left with an upper triangular determinant with a nonzero last row that we want to get rid of in the following. Adding row 1 to row n, the first k elements of row n become zero while the last element becomes $a+b$. Successively adding row $k+1, 2k+1, \dots, \lfloor\frac{n-1}{k}\rfloor k+1$ to the last row, it is possible to eliminate the first $\lfloor\frac{n-1}{k}\rfloor k$ elements of the last row while the last element sums up to $a+\lfloor\frac{n-1}{k}\rfloor b$ and we are left with

$$\begin{vmatrix}
a-b & a-b & \cdots & a-b & 0 & \cdots & 0 & b \\
0 & a-b & a-b & \cdots & a-b & \ddots & \vdots & \vdots \\
\vdots & \ddots & \ddots & \ddots & & \ddots & 0 & b \\
0 & \cdots & 0 & a-b & a-b & \cdots & a-b & b \\
0 & \cdots & 0 & 0 & a-b & \cdots & a-b & 0 \\
\vdots & & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & a-b & 0 \\
0 & \cdots & 0 & 0 & \cdots & -a+b & -a+b & a+\lfloor\frac{n-1}{k}\rfloor b
\end{vmatrix}.$$

4. The remaining nonzero elements of the last row (apart from the last element) can be eliminated by adding row $n-l$ where $n-k \le l \le n-1$ (precisely we have to add row $n-(n-1) \mod k$ if there still are nonzero elements beside the last element of row n). This does not affect the last element of row n, hence we obtained an upper triangular determinant. Since we used only elementary row and column operations, we obtain the formula by multiplying the main diagonal elements and receive

$$|M_n(a,b,[0,1,\dots,k])| = (a-b)^{n-1}(a+\left\lfloor\frac{n-1}{k}\right\rfloor b). \tag{5.15}$$

We imposed the restriction that the a band has to start at position 0 (i.e. starting with the main diagonal). Clearly, $|M_n(a,b,[q,\dots,p])| = 0$ for $2 \le q \le p$, since we obtain a zero column applying the same process as above. A little thought establishes the formula

$$|M_n(a,b,[1,\dots,k])| = (-1)^{n+1}(a-b)^{n-1}b.$$

for arbitrary k and the a band starting at the first upper side diagonal.

What about $|M_n(a,b,[-q,\dots,p])|$?

Assume w.l.o.g. that $q \leq p$. Performing step 1. we get

$$\left|\begin{array}{ccccccccc}
a-b & \cdots & a-b & 0 & \cdots & \cdots & \cdots & 0 & b \\
\vdots & \ddots & & \ddots & \ddots & & & \vdots & \vdots \\
a-b & & \ddots & & \ddots & \ddots & & \vdots & \vdots \\
0 & \ddots & & \ddots & & \ddots & \ddots & \vdots & \vdots \\
\vdots & \ddots & \ddots & & \ddots & & \ddots & 0 & \vdots \\
0 & \cdots & 0 & a-b & \cdots & a-b & \cdots & a-b & b \\
-a+b & \cdots & 0 & \cdots & \cdots & \cdots & \cdots & 0 & a \\
\vdots & & \ddots & \ddots & & & & \vdots & \vdots \\
-a+b & \cdots & & -a+b & 0 & \cdots & \cdots & 0 & a
\end{array}\right|.$$

Comparing this with (5.14), we observe that we would have something similar if we could transform the first q rows appropriately and swap them down to the bottom of the matrix.

We attempt to transform the first q rows from $(a-b, \ldots, a-b, 0, \ldots, 0, b)$ to $(0, \ldots, 0, -a+b, \ldots, -a+b, 0, \ldots, 0)$. To achieve this, we subtract row $q+1$ from row 1, then add row $(p+q+1)+1$, then subtract row $(2q+p+1)+1$ and add row $2(q+p+1)+1$, and so on until we finally subtract row $n-p-1$ to get the desired form. The other rows are treated similarly: Row l is transformed by successively subtracting row $q+1$ and adding row $(p+q+1)+l$, subtracting row $(2q+p+1)+1$ and adding row $(2q+p+1)+l$ until we subtracted row $n-p-1$.

After this transformation, we have

$$\left|\begin{array}{cccccccccc}
0 & \cdots & \cdots & 0 & -a+b & \cdots & -a+b & 0 & \cdots & 0 \\
\vdots & & & \vdots & \ddots & \ddots & \vdots & \vdots & & \vdots \\
0 & \cdots & \cdots & 0 & \cdots & 0 & -a+b & 0 & \cdots & 0 \\
a-b & \cdots & \cdots & \cdots & \cdots & a-b & 0 & \cdots & 0 & b \\
0 & \ddots & & & & & \ddots & \ddots & \vdots & \vdots \\
\vdots & \ddots & \ddots & & & & & \ddots & 0 & b \\
0 & \cdots & 0 & a-b & \cdots & \cdots & \cdots & \cdots & a-b & b \\
-a+b & \cdots & \cdots & -a+b & 0 & \cdots & 0 & \cdots & 0 & a \\
\vdots & & & & \ddots & \ddots & \vdots & & \vdots & \vdots \\
-a+b & \cdots & \cdots & -a+b & \cdots & -a+b & 0 & \cdots & 0 & a
\end{array}\right|.$$

We observe that for $n = k \cdot (p+q+1)+1$, there is only one zero after the $-a+b$ triangular block in the first q rows. The number of right end zeroes increases by one with ascending n until a maximum of $p+q+2$ for $n = (k+1) \cdot (p+q+1)$ (afterwards it drops to one again). This means that the position of the last occurrence of $-a+b$ in the first row coincides with the position of the last occurrence of $-a+b$ in row $n-p$ for $n = k(p+q+1)$ and that the position of the first $-a+b$ occurrence in the first row coincides with the first zero occurrence in the last row for $n = k(p+q+1)+1$. For all other n it is possible to subtract two successive rows of the first q rows from two successive rows of the p last rows, making the resulting rows identical and hence detecting the singularity of the matrix for those values of n.

Let us now focus on the two remaining cases:

Assume that $n = k \cdot (p+q+1)+1$ and notice that in this case, the $-a+b$ triangular block of the first q rows is located exactly above the zero gap in the last p rows. Thus, we add the last row to all of the first q rows. After swapping row $q-l$ down to the bottom for $l = 0, \ldots, q-1$ using $nq - \frac{q(q+1)}{2}$ row exchanges, we get

$$\begin{vmatrix}
a-b & \cdots & \cdots & \cdots & \cdots & \cdots & a-b & 0 & \cdots & 0 & b \\
0 & \ddots & & & & & & \ddots & \ddots & \vdots & \vdots \\
\vdots & \ddots & \ddots & & & & & & \ddots & 0 & b \\
0 & \cdots & 0 & a-b & \cdots & \cdots & \cdots & \cdots & \cdots & a-b & b \\
-a+b & \cdots & \cdots & -a+b & 0 & \cdots & 0 & \cdots & 0 & 0 & a \\
\vdots & & & \vdots & \ddots & \ddots & \vdots & & \vdots & \vdots & \vdots \\
-a+b & \cdots & \cdots & -a+b & \cdots & -a+b & 0 & \cdots & 0 & 0 & a \\
-a+b & \cdots & \cdots & -a+b & \cdots & -a+b & 0 & \cdots & 0 & -a+b & a \\
\vdots & & & \vdots & & \vdots & \vdots & \cdot & \cdot & \vdots & \vdots \\
\vdots & & & \vdots & & \vdots & 0 & \cdot & & \vdots & \vdots \\
-a+b & \cdots & \cdots & -a+b & \cdots & -a+b & -a+b & \cdots & \cdots & -a+b & a
\end{vmatrix}.$$

Now it remains to get the triangular block in the bottom right corner into desired form. Therefore we swap column $n-1$ through to column $n-q$, then the new column $n-1$ through to column $n-q+1$ and so on, using $\frac{q(q-1)}{2}$ column exchanges. (We could achieve the same result with pairwise swapping and $\lfloor \frac{q}{2} \rfloor$ column exchanges but we prefer the previous method since sign factors cancel out). Finally, this yields

$$\begin{vmatrix}
a-b & \cdots & \cdots & a-b & \cdots & a-b & 0 & \cdots & 0 & a-b & b \\
0 & \ddots & & \vdots & & \vdots & \vdots & \cdot & \cdot & \vdots & \vdots \\
\vdots & \ddots & \ddots & \vdots & & \vdots & 0 & \cdot & & \vdots & \vdots \\
0 & \cdots & 0 & a-b & \cdots & a-b & a-b & \cdots & \cdots & a-b & b \\
-a+b & \cdots & \cdots & -a+b & 0 & \cdots & 0 & \cdots & \cdots & 0 & a \\
\vdots & & & \vdots & \ddots & \ddots & \vdots & & & \vdots & \vdots \\
-a+b & \cdots & \cdots & -a+b & \cdots & -a+b & 0 & \cdots & \cdots & 0 & a \\
-a+b & \cdots & \cdots & -a+b & \cdots & -a+b & -a+b & 0 & \cdots & 0 & a \\
\vdots & & & \vdots & & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\
\vdots & & & \vdots & & \vdots & \vdots & & \ddots & 0 & a \\
-a+b & \cdots & \cdots & -a+b & \cdots & -a+b & -a+b & \cdots & \cdots & -a+b & a
\end{vmatrix}, \tag{5.16}$$

which is almost like (5.14) (after performing step 1. on $M_n(a, b, [0, \ldots, p+q+1])$. Observe that performing steps 2. - 4. on (5.16) indeed yields the same triangular determinant. Hence we get the formula

$$|M_n(a, b, [-q, \ldots, p])| = (-1)^{(n-1)q}(a-b)^{n-1}(a+kb). \tag{5.17}$$

for $n = k \cdot (p+q+1) + 1$.

Now we discuss the second case: Assume that $n = k \cdot (p+q+1)$. Since there is a gap of $p+q$ zeroes between the column position of the last $-a+b$ occurrence of the upper $q \times n$ block and the last column, we successively add row $n-p-l$ to row l for $l = 1, \ldots, q$ and get

$$\left|\begin{array}{cccccccccccccccccc}
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \cdots & 0 & a-b & \cdots & a-b & a-b & 0 & \cdots & 0 & b \\
\vdots & & & & & & & \vdots & & \vdots & \vdots & & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\
\vdots & & & & & & & \vdots & & \vdots & \vdots & & \vdots & \vdots & & \ddots & 0 & b \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \cdots & 0 & a-b & \cdots & a-b & a-b & \cdots & \cdots & a-b & b \\
a-b & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & a-b & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & b \\
0 & \ddots & & & & & & & \ddots & \ddots & & & & & & & \vdots & \vdots \\
\vdots & \ddots & \ddots & & & & & & & \ddots & \ddots & & & & & & \vdots & \vdots \\
\vdots & & \ddots & \ddots & & & & & & & \ddots & \ddots & & & & & \vdots & \vdots \\
\vdots & & & \ddots & \ddots & & & & & & & \ddots & \ddots & & & & \vdots & \vdots \\
\vdots & & & & \ddots & \ddots & & & & & & & \ddots & \ddots & & & \vdots & \vdots \\
\vdots & & & & & \ddots & \ddots & & & & & & & \ddots & \ddots & & \vdots & \vdots \\
\vdots & & & & & & \ddots & \ddots & & & & & & & \ddots & \ddots & \vdots & \vdots \\
\vdots & & & & & & & \ddots & \ddots & & & & & & & \ddots & 0 & b \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & a-b & & & & & & & a-b & b \\
b-a & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & b-a & b-a & 0 & \cdots & \cdots & 0 & \cdots & \cdots & 0 & a \\
\vdots & & & & & & & & & \vdots & \ddots & \ddots & & \vdots & & & \vdots & \vdots \\
\vdots & & & & & & & & & \vdots & & \ddots & \ddots & \vdots & & & \vdots & \vdots \\
b-a & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & b-a & b-a & \cdots & \cdots & b-a & 0 & \cdots & \cdots & 0 & a
\end{array}\right|.$$

Then we add row $n-q$ to the the negative of row l for $l=1,\ldots,q$, introducing a signfactor $(-1)^q$ and get

$$\left|\begin{array}{cccccccccccccccc}
-a+b & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & -a+b & -a+b & 0 & \cdots & 0 & a-b \\
\vdots & & & & & & & & & & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\
\vdots & & & & & & & & & & \vdots & \vdots & & \ddots & 0 & \vdots \\
-a+b & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & -a+b & -a+b & \cdots & \cdots & -a+b & a-b \\
a-b & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & a-b & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & b \\
0 & \ddots & & & & & & & \ddots & \ddots & & & & & \vdots & \vdots \\
\vdots & \ddots & \ddots & & & & & & & \ddots & \ddots & & & & \vdots & \vdots \\
\vdots & & \ddots & \ddots & & & & & & & \ddots & \ddots & & & \vdots & \vdots \\
\vdots & & & \ddots & \ddots & & & & & & & \ddots & \ddots & & \vdots & \vdots \\
\vdots & & & & \ddots & \ddots & & & & & & & \ddots & \ddots & \vdots & \vdots \\
\vdots & & & & & \ddots & \ddots & & & & & & & \ddots & 0 & b \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & a-b & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & a-b & b \\
-a+b & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & -a+b & 0 & \cdots & 0 & 0 & \cdots & \cdots & 0 & a \\
\vdots & & & & & & & \vdots & \ddots & \ddots & \vdots & \vdots & & & \vdots & \vdots \\
\vdots & & & & & & & \vdots & & \ddots & 0 & \vdots & & & \vdots & \vdots \\
-a+b & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & -a+b & \cdots & \cdots & -a+b & 0 & \cdots & \cdots & 0 & a
\end{array}\right|.$$

Next, we swap row l down to the bottom for $l=1,\ldots,q$ using $(n-1)q$ row exchanges. This resembles (5.14) for $M_n(a,b,[0,\ldots,p+q+1)$ apart from entries $a-b$ instead of a in the last q rows. After performing step 2. we just differ in the last element of the last $p+q$ rows where we have $a-b$ instead of a in the last row and $-b$ instead of 0 in the others. Since $n = k(p+q+1)$ we have to perform step 4. after step 3. and get the product of the main diagonal elements

$$|M_n(a,b,[-q,\dots,p])| = (a-b)^{n-1}(a + \left\lfloor \frac{n-1}{p+q+1} \right\rfloor b).$$

Since $n = k \cdot (p+q+1)$ and taking the sign factor into account, we end up with the determinant formula

$$|M_n(a,b,[-q,\dots,p])| = (-1)^{nq}(a-b)^{n-1}(a+(k-1)b).$$

Recalling that transposing the matrix would remove the restriction $q \leq p$, we have the following theorem.

Theorem 27 *Let $M_n(a,b,[-q,\dots,p])$ be a two element diagonal matrix of order n with $q,p \in \mathbb{N}$. The determinant formula is*

$$|M_n(a,b,[-q,\dots,p])| = \begin{cases} (-1)^{(n-1)q}(a-b)^{n-1}(a+kb), & \mathit{if} n = k(p+q+1)+1 \\ (-1)^{nq}(a-b)^{n-1}(a+(k-1)b), & \mathit{if} n = k(p+q+1) \\ 0, & \mathit{otherwise.} \end{cases}$$

Closing, we will briefly elaborate on the question: How does the determinant change if we consider a bands with "holes", i.e. what happens to the general problem $|M_n(a,b,[p_1,\dots,p_k)|$ with $p_1 \leq \dots \leq p_k$?

We will investigate the case that $p_1 = 0$. (Obviously $|M| = 0$ for $p_1 > 1$ and $|M| = (-1)^{(n+1)}(a-b)^{n-1}b$ for $k_1 = 1$).

After performing step 1. we receive something like

$$\begin{vmatrix} a-b & 0 & a-b & 0 & a-b & 0 & \cdots & 0 & b \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & a-b & 0 & a-b & 0 & a-b & b \\ -a+b & \cdots & -a+b & -a+b & 0 & -a+b & 0 & -a+b & a \\ 0 & \cdots & 0 & 0 & 0 & a-b & 0 & a-b & b \\ -a+b & \cdots & -a+b & -a+b & -a+b & -a+b & 0 & -a+b & a \\ 0 & \cdots & 0 & 0 & 0 & 0 & 0 & a-b & b \\ -a+b & \cdots & -a+b & -a+b & -a+b & -a+b & -a+b & -a+b & a \end{vmatrix}.$$

It is possible to transform this into triangular form with a nonzero last row by subtracting the last row from row $n - p_l$ for $l = 2,\dots,k$ which leaves us with something like

$$\begin{vmatrix} a-b & 0 & a-b & 0 & a-b & 0 & \cdots & 0 & b \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & a-b & 0 & a-b & 0 & a-b & b \\ 0 & \cdots & 0 & 0 & a-b & 0 & a-b & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 & a-b & 0 & a-b & b \\ 0 & \cdots & 0 & 0 & 0 & 0 & a-b & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & 0 & a-b & b \\ -a+b & \cdots & -a+b & -a+b & -a+b & -a+b & -a+b & -a+b & a \end{vmatrix}.$$

Unfortunately it is not possible to simply apply step 3. and 4. because of the holes. Thus, we try to transform the matrix into arrow form by subtracting row $n-1$ from row $n-1-p_l$, for $l = 2,\dots,k$, then row $n-2$ from row $n-2-p_l$ for $l = 2,\dots,k$ and so on until we reached arrow form. Now, since the first

$n-1$ elements of the main diagonal are $a-b$ and the first $n-1$ elements of the last row are $-a+b$, we simply have to add row l to row n for $l=1,\dots,n-1$ to reach triangular form. However, in general it is not possible to find a defining formula for the resulting last element of the last row and hence the determinant.

After investigating determinant formulas of two element Toeplitz matrices we will show how a special class of Toeplitz matrices can be transformed into frame form which means that its determinant formulas can be determined using the theorems of the second chapter.

Consider determinants of Toeplitz matrices of the form $A(n,[y,b,\dots,b,a,c,b,\dots,bz])$:

$$|A| = \begin{vmatrix} a & c & b & \cdots & b & z \\ b & \ddots & \ddots & \ddots & & b \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & b \\ b & & & \ddots & \ddots & c \\ y & b & \cdots & \cdots & b & a \end{vmatrix}.$$

Subtracting the first column from all remaining columns it is possible to transform the Toeplitz determinant into fat R–form

$$|A| = \begin{vmatrix} a & c-a & b-a & \cdots & b-a & z-a \\ b & a-b & c-b & 0 & \cdots & 0 \\ b & 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & 0 \\ b & 0 & \cdots & 0 & a-b & c-b \\ y & b-y & \cdots & b-y & b-y & a-y \end{vmatrix},$$

thus we get a determinant formula using results from the second chapter. We omit the resulting formula since it is quite involved and point out the possibility to obtain it using the corresponding package function `Rform`.

Closing we will show how it is possible to compute determinant formulas for a certain class of Toeplitz matrices reducing them to simple Hessenberg matrices. Consider determinants of Toeplitz matrices of the form $A(n,[b,\dots,b,a,c,\dots,c,u_1,u_2,\dots,u_p])$ with integer p.

$$|A| = \begin{vmatrix} a & c & \cdots & c & u_1 & \cdots & u_{p-1} & u_p \\ b & \ddots & \ddots & & \ddots & \ddots & & u_{p-1} \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & & \ddots & u_1 \\ \vdots & & & \ddots & \ddots & \ddots & & c \\ \vdots & & & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & & & \ddots & \ddots & c \\ b & \cdots & \cdots & \cdots & \cdots & \cdots & b & a \end{vmatrix}$$

Performing the operations $col_k = col_k - col_{k+1}$ for $k=1,\dots,n-1$ we may reduce $|A|$ to

$$|A| = \begin{vmatrix}
a-c & 0 & \cdots & 0 & c-u_1 & u_1-u_2 & \cdots & u_{p-2}-u_{p-1} & u_{p-1}-u_p & u_p \\
b-a & \ddots & \ddots & & \ddots & \ddots & \ddots & & u_{p-2}-u_{p-1} & u_{p-1} \\
0 & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \vdots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & & \ddots & \ddots & u_1-u_2 & u_2 \\
\vdots & & \ddots & \ddots & \ddots & \ddots & & \ddots & c-u_1 & u_1 \\
\vdots & & & \ddots & \ddots & \ddots & \ddots & & 0 & c \\
\vdots & & & & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\
\vdots & & & & & \ddots & \ddots & \ddots & 0 & c \\
\vdots & & & & & & \ddots & \ddots & a-c & c \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & b-a & a
\end{vmatrix},$$

which is a Hessenberg determinant. Expanding the last row like in the proof of (4.9) yields

$$\begin{aligned}
|A| \;=\; & \mathrm{HB}(n-1,[b-a,a-c,0,\ldots,0,c-u_1,u_1-u_2,\ldots,u_{p-1}-u_p]) \qquad (5.18)\\
& +\sum_{l=1}^{p-1}(-1)^l c\cdot(b-a)^l \mathrm{HB}(n-l-1,[n-a,a-c,0,\ldots,0,c-u_1,u_1-u_2,\ldots,u_{p-l-1}-u_{p-l}])\\
& +\sum_{l=p}^{n-p}(-1)^l c\cdot(b-a)^l(a-c)^{n-l-1}+\sum_{l=n-p+1}^{n}(-1)^l u_{n+p-l}(b-a)^l(a-c)^{n-l-1}.
\end{aligned}$$

Recall that in (4.10) we were able to derive a formula for the type of Hessenberg determinant arising in this computation.

Analogous proceeding for Toeplitz matrices of the form $A(n,[b,\ldots,b,a,\ldots,a,c,\ldots,c,u_1,\ldots,u_p])$ that have an a band of fixed integer length q yields

$$\begin{vmatrix}
0 & \cdots & 0 & a-c & 0 & \cdots & 0 & c-u_1 & u_1-u_2 & u_2-u_3 & \cdots & u_{p-1}-u_p & u_p \\
b-a & \ddots & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \ddots & \vdots & u_{p-1} \\
0 & \ddots & \ddots & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & u_2-u_3 & \vdots \\
\vdots & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & \ddots & \ddots & u_1-u_2 & u_2 \\
\vdots & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & \ddots & c-u_1 & u_1 \\
\vdots & & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & 0 & c \\
\vdots & & & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \vdots & \vdots \\
\vdots & & & & & \ddots & \ddots & \ddots & & \ddots & \ddots & 0 & c \\
\vdots & & & & & & \ddots & \ddots & \ddots & & \ddots & a-c & c \\
\vdots & & & & & & & \ddots & \ddots & \ddots & & 0 & a \\
\vdots & & & & & & & & \ddots & \ddots & \ddots & \vdots & \vdots \\
\vdots & & & & & & & & & \ddots & \ddots & 0 & a \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & b-a & a
\end{vmatrix}.$$

Expanding the last row like in the proof of (4.9) again and assuming $q = p$ for simplicity, we receive

$$\begin{aligned}|A| &= \sum_{l=0}^{q-1}(-1)^l a\cdot(b-a)^l \mathrm{HB}(n-l-1,[b-a,0,\dots,0,a-c,0,\dots 0,c-u_1,u_1-u_2,\dots,u_{p-l-1}-u_{p-l}]) \\ &+\sum_{l=q}^{n-p}(-1)^l c(b-a)^l \mathrm{HB}(n-l-1,[b-a,0,\dots,0,a-c,0,\dots,0]) \\ &+\sum_{l=n-p+1}^{n}(-1)^l u_{n+p-l}(b-a)^l \mathrm{HB}(n-l-1,b-a,0,\dots,0]).\end{aligned}$$

Note that we found a formula for the type of the occurring Hessenberg determinants in (4.12) and (4.15). However, the formula involved two cases depending on the determinant order. This distinction of two cases leads to a $q+1$ case distinction for $|A|$. Note that the Hessenberg determinants in the second and the third sum are either zero or have a nice closed form.

The case $q \neq p$ is more clumsy but also results in a $q+1$ case formula. We omit the details pointing out that the implemented Maple function `ToeplitzDet` also deals with this case.

5.4.2 Circulants

We discuss another special form of persymmetric determinants following [Met60].

Definition 27 *A $n\times n$ matrix A is called a* circular matrix *if any row is a "rightshifted" version of its preceding row. We will refer to the determinant of a circular matrix as* circulant. *If the first row of C is $a_1,\dots,a_n$ we will denote the circulant as $C(a_1,\dots,a_n)$.*

Example

$$C(a_1,\dots,a_n) = \begin{vmatrix} a_1 & a_2 & \ddots & a_{n-1} & a_n \\ a_n & a_1 & a_2 & \ddots & a_{n-1} \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ a_3 & \ddots & \ddots & \ddots & a_2 \\ a_2 & a_3 & \ddots & a_n & a_1 \end{vmatrix}.$$

A circulant is evidently a special case of a Toeplitz determinant (with diagonal generating functions $a_2,\dots,a_n$, $a_1,\dots,a_n$) and therefore persymmetric.

It is frequently more convenient to define a circulant as a determinant such that any row is the "leftshifted" version of its predecessor. We will denote such circulants as $C'(a_1,\dots,a_n)$. By transposition of the rows it appears that

$$C'(a_1,\dots,a_n) = (-1)^{(n-1)(n-2)/2}C(a_1,\dots,a_n). \tag{5.19}$$

We will now present a fundamental theorem concerning circulants.

Theorem 28 *Consider the circulant $C(a_1,\dots,a_n)$. The following formula holds*

$$C(a_1,\dots,a_n)=\prod_{k=1}^{n}(a_1+\xi_k a_2+\xi_k^2 a_3+\cdots+\xi_k^{n-1}a_n), \tag{5.20}$$

where

$$\xi_k=\exp(2\pi i\cdot k/n)$$

is a complex nth root of unity.

Proof.

Let $\xi_k=\exp(2\pi i\cdot k/n)$ and Δ be the following alternant

$$\Delta=\begin{vmatrix} 1 & 1 & \cdots & 1\\ \xi_1 & \xi_2 & \cdots & \xi_n\\ \vdots & \vdots & & \vdots\\ \xi_1^{n-1} & \xi_2^{n-1} & \cdots & \xi_n^{n-1}\end{vmatrix}.$$

Multiplying $C=C(a_1,\dots,a_n)$ from the right with Δ we get

$$C\cdot\Delta=\begin{vmatrix} a_1+a_2\xi_1+\cdots+a_n\xi_1^{n-1} & a_1+a_2\xi_2+\cdots+a_n\xi_2^{n-1} & \cdots & a_1+a_2\xi_n+\cdots+a_n\xi_n^{n-1}\\ a_2+a_3\xi_1+\cdots+a_1\xi_1^{n-1} & a_2+a_3\xi_2+\cdots+a_1\xi_2^{n-1} & \cdots & a_2+a_3\xi_n+\cdots+a_1\xi_n^{n-1}\\ \vdots & \vdots & & \vdots\\ a_n+a_1\xi_1+\cdots+a_{n-1}\xi_1^{n-1} & a_n+a_1\xi_2+\cdots+a_{n-1}\xi_2^{n-1} & \cdots & a_n+a_1\xi_n+\cdots+a_{n-1}\xi_n^{n-1}\end{vmatrix}.$$

Since $\xi_k^p=\xi_k^{p \bmod n}$ and $\xi_k^{-p}=\xi_k^{n-p \bmod n}$ for positive p, and using θ_k to represent $a_1+a_2\xi_k+\cdots+\xi_k^{n-1}$ this simplifies to

$$C\cdot\Delta=\begin{vmatrix} \theta_1 & \theta_2 & \cdots & \theta_n\\ \theta_1\xi_1 & \theta_2\xi_2 & \cdots & \theta_n\xi_n\\ \vdots & \vdots & & \vdots\\ \theta_1\xi_1^{n-1} & \theta_2\xi_2^{n-1} & \cdots & \theta_n\xi_n^{n-1}\end{vmatrix}=\theta_1\theta_2\cdots\theta_n\cdot\Delta$$

and the theorem follows.

□

It may be observed that every circulant contains $a_1+a_2+\cdots+a_n$ as a factor and that circulants of even order also contain the factor $a_1-a_2+a_3-\cdots+a_{n-1}-a_n$.

Following we will take a look at several special circulants and show how we can obtain their specific formulas.

Corollary 16 *Consider the circulant $C(a,\dots,a,b,\dots,b)$ of order n, where p is the number of a's and q is the number of b's in a row (i.e. $p+q=n$). The following formula holds.*

$$C(a,\dots,a,b,\dots,b)=\begin{cases}(pa+qb)(a-b)^{n-1}, & gcd(p,q)=1\\ 0, & otherwise.\end{cases} \tag{5.21}$$

Proof.

Denoting $C = C(a, \dots, a, b, \dots, b)$ and applying (5.20) we get

$$C = \prod_{k=1}^{n} a(1 + \xi_k + \cdots + \xi_k^{p-1}) + b(\xi_k^p + \cdots + \xi_k^{n-1}). \tag{5.22}$$

Choosing $k = n$ we obtain the factor $pa + qb$. Since $1 + \xi_k + \cdots + \xi_k^{n-1} = 0$ we may rewrite (5.22) as

$$C = (pa + qb) \prod_{k=1}^{n-1} (a - b)(1 + \xi_k + \cdots + \xi_k^{p-1}) = (pa + qb)(a - b)^{n-1} \prod_{k=1}^{n-1} (1 + \xi_k + \cdots + \xi_k^{p-1}).$$

It remains to show that $P := \prod_{k=1}^{n-1}(1+\xi_k+\cdots+\xi_k^{p-1})$ is 1 if $\gcd(p, q) = 1$ and 0 otherwise. We may rewrite P as

$$P = \prod_{k=1}^{n-1} \frac{\xi_k^p - 1}{\xi_k - 1}.$$

If $\gcd(p, q) = 1$ then it follows that $\gcd(n, p) = 1$ since $n = p + q$ and thus ξ_1^p is also a primitive root of unity (we have $\xi_k^p \neq 1$ for $k = 1, \dots, n - 1$). Hence, we get all distinct roots of unity subtracted by 1 in the nominator and the denominator, apart from a different order; they cancel out and we are left with 1.

If $\gcd(p, q) \neq 1$ then $\gcd(n, p) \neq 1$ which means that $\xi_k^p = 1$ for a k between 1 and $n - 1$. Hence the product becomes zero.

□

Note that the corollary extends to circulants of the form $C(a, \dots, a, b, \dots, b, a, \dots, a)$ since it is possible to transform them into $\overline{C}(a, \dots, a, b, \dots, b)$ if we are successively swapping the first row down to the bottom.

Next, we will investigate circulants that are generated "vandermonde" like.

Corollary 17 *Consider circulants of the form $C(1, x, x^2, \dots, x^{n-1})$, then we have the formula*

$$C(1, x, x^2, \dots, x^{n-1}) = (1 - x^n)^{n-1}. \tag{5.23}$$

Proof.

Perform the elementary column operations $col_k = col_k - x \cdot col_{k-1}$ for $k = n, n-1, \dots, 2$ and observe that in every step we subtract x^n from the corresponding main diagonal position while reducing the other column elements to zero.

□

Obviously, this corollary allows the generalization to use $q(x) \cdot p(x)^{l(i)}$ as generating functions instead of x^{i-1} for polynomial functions q and p and a linear function l.

We will close with another special circulant generated by the function $f(i) = a + (i - 1)d$.

Corollary 18 *Consider the circulant $C(a, a + d, a + 2d, \dots, a + (n - 1)d)$. The following formula holds*

$$C(a, a + d, a + 2d, \dots, a + (n - 1)d) = (-1)^{n-1}(dn)^{n-1}(a + (n - 1)d/2). \tag{5.24}$$

Proof.

Denote $C = C(a, a+d, \ldots, a+(n-1)d)$. If we add all the rows to the first row, it appears that $na+n(n-1)d/2$ is a factor. Removing this factor and performing the elementary column operations $col_k = col_k - col_1$ for $k = 2, \ldots, n$ we get

$$C = (na + n(n-1)d/2)\begin{vmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ a+(n-1)d & -(n-1)d & -(n-2)d & \cdots & -2d & -d \\ a+(n-2)d & d & -(n-2)d & \cdots & -2d & -d \\ \vdots & \vdots & 2d & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & -2d & -d \\ a+2d & d & 2d & \cdots & (n-2)d & -d \end{vmatrix}.$$

We observe that $-d$ is a factor of all but the first column. Taking this out and expanding the first row, we are left with

$$C = (na + n(n-1)d/2)(-d)^{n-1}\begin{vmatrix} n-1 & n-2 & \cdots & 2 & 1 \\ -1 & n-2 & \cdots & 2 & 1 \\ \vdots & -2 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 2 & 1 \\ -1 & -2 & \cdots & -n-2 & 1 \end{vmatrix}_{n-1}.$$

Performing the operations $row_k = row_k - row_{k+1}$ for $k = 1, \ldots, n-1$, we see that the determinant on the right has the formula n^{n-2}. The theorem follows.

□

5.5 Implementation

We address implementation issues of the Maple package SYMMETRIC for the discussed symmetric determinants. Examples and further details can be found in the appendix or the on–line help pages.

5.5.1 General considerations

Determinant order The determinant order can be a positive integer or a symbolic expression of the form $n + d$ with integer d. However, in the centrosymmetric case we only allow integer orders.

How do we specify the determinant ? Since symmetric determinants are still very general and we were only able to find formulas for special cases, we only allow quite restrictive specifications in the case of centrosymmetric and axisymmetric determinants. The specification of Toeplitz determinants and Circulants which have a more special structure can be more general. See the corresponding package functions for details.

Checking the determinant formula The computed formulas can be checked for integer orders (default 4) like in the preceding packages.

5.5.2 The package functions

The SYMMETRIC package consists of the following functions:

`AxisymmetricDet, AxisymmetricMatrix, CentrosymetricDet, CentrosymmetricMatrix,`

`Circulant, CirculantMatrix, ToeplitzDet, ToeplitzMatrix.`

CentrosymmetricDet We omitted the derived formulas for symbolic order since they originated in non–centrosymmetric determinants. Thus, we only allow integer orders and a restrictive generating function $a(i)$ and compute the determinant using the methods of the first section.

CentrosymmetricMatrix This function returns the specified centrosymmetric matrix of integer dimension.

AxisymmetricDet We only allow a function in i specifying the main diagonal and a function $restF$ in i and j specifying the remaining elements. If $restF$ is constant we determine the arrow form and return its formula. If $restF$ is only a function in i or in j we determine the transformation into continuant form and return the corresponding recurrence relation. It is not possible to obtain a formula for the most general case, except if $a_{ij} = c(j-i)+d$ or $a_{ij} = c(i+j)+d$ (see (5.7)).

If the optional directive skew is given, we compute the square of the corresponding Pfaffian according to (5.12) for integer orders. In the symbolic case, we return a recurrence relation if $restF$ is only a function in i or only in j; For constant $restF$ we determine the corresponding 7–form determinant. If the case $a_{ij} = c(i-j)+d$ or $a_{ij} = c(i+j)+d$ is detected, we determine the corresponding frame form determinant and call the appropriate FRAMEFORMS function.

AxisymmetricMatrix This function returns the specified axisymmetric (or skewaxisymmetric) matrix. Dots "o" are used to abbreviate the symbolic case. Since Maple treats the dots as usual matrix entries, this function should be used for illustrative purposes only for symbolic orders.

ToeplitzDet The specification of the Toeplitz determinant is a piecewise definition of the constant diagonals, i.e. a partition of the interval [-n+1..n-1]. If there is a two element band, we return the corresponding formula according to (5.15) and Theorem 27. Otherwise we try to detect the cases where transformation into Hessenberg form or into frame form is possible, determine the corresponding specifications and return the formula resulting from the corresponding function calls.

ToeplitzMatrix This function returns the specified Toeplitz matrix. The same remarks as above apply.

Circulant The specification of a circulant is the piecewise specification of its first row (like in the FRAMEFORMS package). The remaining rows are obtained by right shifts of the preceding rows. If the optional directive leftshift is given, we have the other construction method and retransform it into the preceding using (5.19). If the circulant is generated by $a + id$ we return the formula (5.24). For a generating function of the form $q(x)p(x)^{l(i)}$ we return formula (5.23). If we detect two element band form we use (5.21) to obtain a formula.

In the general case we compute the general formula (5.20) involving roots of unity. Unfortunately Maple has problems with the manipulation of these roots of unity so that it is not possible to compute a circulant of odd order, we have to settle with the unevaluated general formula in that case.

CirculantMatrix This function returns the matrix of the specified circulant. The same remarks as above apply

5.6 Summary

In this chapter we discussed determinants of a number of symmetric matrix classes. Since symmetry still yields a quite general form, we only managed to obtain determinant formulas of special cases of these classes. We could derive formulas for a number of special cases of axisymmetric and skewaxisymmetric determinants reducing them to simpler frame form or continuant form. The special structure of a zero axial skew determinant was pointed out including the interesting relation to matching theory. In the case of persymmetric determinants we were able to find formulas for several special cases of Toeplitz determinants using elementary row and column operations to reduce them to simpler forms like frame form and sparse Hessenberg form. We presented an old general determinant formula for circulants and picked out a few special cases of [Met60] where it was possible to find simpler formulas.

The implemented Maple package SYMMETRIC allows the computation of most of the discussed formulas. Using the package it is possible to obtain formulas for many of the exercises in linear algebra books like [Bri83][Jän91][SW89].

Summary and Discussion

In this work we have studied a number of important matrix classes and tried to derive determinant formulas for them. The results were integrated in the computer algebra system Maple by developing packages of functions that enabled the specification of a matrix of a special class followed by the computation of its specific determinant formula. Our aim was to theoretically obtain very general formulas such that it was possible to automatically compute formulas for specified special cases. A large portion of the presented theoretical results were taken from the excellent old book of Thomas Muir which was revised and enlarged by William Metzler [Met60]. We tried to adapt their results to a more modern notation and elaborated the proofs in a little more detail.

We were able to derive determinant formulas for matrices that have at most two nonzero rows and columns and a nonzero maindiagonal. These so–called frame forms frequently arose in later chapters. Moreover, some special geometric predicates can be formulated as frame form determinants for which we can find formulas using the functions of our package. The major drawback of the FRAMEFORMS package is that the resulting determinant formulas are sometimes far from cleverly simplified and possible "nice" formulas can only be revealed after manual simplification. This is partly due to Maple's inability to entirely simplify an expression and partly due to the generalized computation. For example, special cases that are better treated with another simpler method, are computed in the standard way leading to sometimes enormous structural overhead.

The use of results in [Met60] made it possible to obtain nice formulas for special simple alternants expressing them as a product of the difference product of their variables and the corresponding symmetric cofactor. The symmetric cofactor could be represented using elementary symmetric functions or complete symmetric functions and thus clarifying its structure. Unfortunately it was not possible to derive general explicit formulas for polynomial alternants of higher orders. We also could only handle a very restricted class of double alternants by reducing their elementary symmetric function determinant representation to frame form.

In the case of tridiagonal and Hessenberg matrices we derived general recurrence formulas for the determinant and extended some of the special results for continuants in [Met60] to Hessenberg determinants. The resulting formulas were quite involved and required us to restrict the specification of Hessenberg matrices to total diagonal generating functions in our implementation.

The investigation of symmetric matrices lead to a few generalizations of special axisymmetric determinants and special Toeplitz determinants using row and column operations to reduce them to frame form or simple Hessenberg form. We also used results from [Met60] to obtain formulas for circulants and zero–axial skewsymmetric determinants. The implementation made heavy use of our other packages and enabled us to obtain determinant formulas for a lot of examples of special matrices in linear algebra text books (in fact the generalization of many of them was motivated by the induction exercises in these books).

So far we always tried to prove a general theoretical formula for a matrix class and used the theoretical result or method in our implementation. This had the effect that sometimes, especially for symmetric matrices, the software package looked more like a library of determinant formulas. Moreover, the packages are restricted to the covered matrix classes. It would be desirable to have a program that is also able to apply the techniques used to theoretically derive determinant formulas to find new formulas for other matrix classes. This problem is very tricky, however. How do we specify the determinants? Since our techniques can only be fruitful for specially structured matrices, we always have a different specification scheme in mind. How can we write a program that does clever expansion and restructuring which was necessary to prove most of the derived formulas? What we could do is to implement a number of strategies resulting from our own experiences, but finding new formulas does not seem very promising apart from generalizations of frame forms for example.

In this work, we focused on the investigation of determinant formulas of only four important matrix classes. It is unnecessary to note that there is a variety of other interesting classes of specially structured matrices like Wronski matrices occurring in analysis or resultants occurring in algebra, just to name a few, which are not covered here.

Appendix A

The Computer Algebra System Maple

In this chapter we will give a brief overview about the computer algebra system Maple. We will outline the general organization of Maple following [Kal], describe some details of its programming language and the package formalism. Moreover we will discuss, why Maple is especially suitable to implement our determinant system.

More detailed informations can be found in [CGG+92] [Mon], the very good on–line documentation or on the WWW (try for example the Maple page of the University of Waterloo at `http://www.daisy.uwaterloo.ca/`, the commercial site at `http://www.maplesoft.com/` or the excellent Maple V page at

`http://SunSITE.informatik.rwth-aachen.de/maple/maplev.html)` .

A.1 What is Maple ?

Maple is a system for symbolic mathematical computation which has been under development at the University of Waterloo, Canada, since the end of 1980 (see [CGGG83] [CGG+92]). It has its own Algol68–like programming language which enables the user to carry out complicated or repetitive operations and develop their own extension of the system.

In fact most of Maple's own procedures are written in this language. Only a small kernel part in written in the base language C.

The primary design goals of the Maple system were

- compactness
- a powerful set of facilities for mathematical computation
- a good user interface (i.e. fairly natural computation)

The system is highly portable and is available for a large number of machines under various operating systems. This level of portability has been achieved by having only one source code which is written by means of a general purpose macro–processor, called Margay resembling C's macro–processor but even more powerful. The target language is C and the operating system must support a C compiler. The Margay macros reflect machine dependencies, operating system dependencies from one C compiler to the next.

A.1.1 Maple's internal organization

We attempt to give a brief overview about the organization and the design of the Maple system.

The Maple language has a very simple syntax. The main program reads input, calls the parser and then the statement evaluator for each complete statement. An internal data structure is created for each successful production. Maple stops when it evaluates one of `done` , `stop` or `quit`.

A.1.1.1 Internal Functions

The functions which are part of Maple's kernel, called internal functions, fall into four distinct groups.

Evaluators These split into the following functions and associated tasks:

`evalstat` for statement evaluation,

`eval` for expression evaluation,

`evalname` for name formation,

`evalf` for floating point arithmetic.

The parser calls only the function `evalstat` which usually initiates many interactions between the different evaluators. Although users don't have to call any evaluator directly, they may do so if the situation demands it.

Algebraic functions These are identified with functions which are generally available to the user and are called basic functions. Examples include:

`diff` for derivatives,

`op` for selecting parts of expressions,

`divide` for polynomial division,

`coeff` for finding coefficients of polynomials,

`subs, subsop` for substitution of expressions,

`expand` for finding expansion of expressions.

Algebraic service functions These functions cannot normally be called by the user directely. They are at a level lower than the functions of the previous group and are used in their implementation. Examples include the arithmetic packages, the basic simplifier, printing, the set operations package etc.

General service functions These are at the lowest level and may be called by any other function of the system. They are not necessarily tied to algebraic computation but also provide services such as: storage allocation, garbage collection, error–handlers, etc.

A.1.1.2 The Maple Library

Maple's functions fall into four categories:

1. Built–in function which are internal to the system.
2. Library functions which are automatically loaded.
3. Miscellaneous library functions which are loaded by the user via the `readlib` function.

4. Packages of functions which are normally loaded via the `with` function.

The internal functions can and often do call upon functions which reside outside the kernel and are found in the Maple library. For example `expand` does most of the work itself but when expanding a sum to a larger power it will call the library function `‘expand/bigpow‘`.

This feature has the following advantages:

- The system is very flexible since the library can be changed easily,
- Users can define their own handling functions so that personal tailoring is possible,
- The library source code can be read by all users. Using

  ```
  interface(verboseproc=2); print(<function>);
  ```

  it is possible to read the source code for a non–kernel procedure.
- The basic system which is always loaded can stay quite small even though the complex Maple system is very large.

The call `readlib( <procname> )` causes Maple to load a file called `<procname>.m` which is located in the Maple library directory. (Note that files with the `.m` extension are assumed by Maple to contain code in fast loading internal format).

Packages allow the user to define an entire collection of functions with a simple command. After executing `with( <packagename> )` we can use any of the package functions with their short name.

Maple packages load very quickly because the system just reads in a table of pointers where the code is to be found. The actual code is not pulled into memory until it is required.

Packages are the heart of Maple, offering functions for all different areas of symbolic mathematic computation.

Package examples are: `linalg, stats, logic, combinat` etc.

The nice encapsulated form of functionality hidden in packages can also be programmed by the user. We will see later how this works.

A.1.1.3 Internal Representation of Data Types

All internal data structures used by Maple are built by the parser and some internal functions. These structures have the same general format.

header	$data_1$	$data_2$	$\dots$	$data_n$	

The *header* encodes the following information:

- 15 bits giving the length of the structure,
- 6 bits giving its type,
- 1 bit indicating its simplification status,
- 7 bits giving information to the garbage collector.

The *data* items are usually pointers to other internal structures.

The length of a structure does not change and structures are never changed during execution since it is not known which other structures point to them. The normal method of modifying a structure is by making a copy of it and modifying that. Otherwise the only safe modifications are carried out by the simplifier which produces the same value in simpler form. The garbage collector identifies and removes unused structures.

The internal representation of all data structures is by means of dynamic arrays (vectors). This means that each (top level) component part of the data structure an be accessed in constant time and this makes various operations faster than they would be if linked lists were used. Dynamic vectors are also more economical than linked lists in terms of space requirements.

A.1.2 The Maple programming language

We try to give a short description about programming in Maple. Details can be found in [Mon] and [Red94] for example.

Basic language features As mentioned above, Maple has its own programming language. Most of its constructs are borrowed from other languages, e.g. the repetition and selection statements come from Algol68. The language also has a functional flavour in that every procedure returns a result.

Maple is not strongly typed like C or Pascal. No declarations are required (thus Maple is more like Lisp and Basic in this aspect). However, types exist. Type checking is done at run time and must be programmed explicitly.

Maple is interactive and the programming language is interpreted. Maple is not suitable for running numerically intensive programs because of the interpreter overhead.

The heart of Maple programming is writing procedures that can make use of all Maple functions. A typical procedure definition looks like this:

```
proc(<name sequence>)
  [ local <name sequence>; ]
  [ options <name sequence>; ]
  <statement sequence>
end;
```

Example: `squaresum:=proc(x,y) x*x+y*y end:`

Calling this defined procedure would look like that: `squaresum(2,3);` with result `13.`

Each procedure is given a (possibly empty) parameter list. The parameter passing mechanism can be described as 'call by evaluated name'. All actual parameters are first evaluated and then each formal parameter is replaced by its corresponding actual parameter (and is not evaluated again). If we want to pass a value back via a parameter we have to make sure that at the time of the call the parameter has no value assigned to it.

It is possible to declare local arguments of a procedure. They can only be seen within this procedure (and not in possible nested procedures). However, it is not necessary to declare any variable used in a procedure. If declaration is omitted, the variable is assumed to be global.

Local variables are only given one level of evaluation, as opposed to global ones which are given fully recursive evaluation. This is for efficiency reasons.

Parameter passing and scope rules are somewhat unusual and can cause nasty surprises if one is not aware of them.

A very useful facility in Maple procedures is the availability of various options, the most important being the option `remember`. This causes Maple to store the results of calls to a particular procedure so that in future invocations they will just be looked up rather than recomputed. A good example is the computation of the Fibonacci numbers. The clearest code is obtained by implementing the recurrence which defines them. Unfortunately this is incredibly inefficient. The use of the `remember` option allows us to retain the clear code and compute the numbers efficiently.

Data types in Maple Each Maple expression is represented internally by a tree each of its vertices is some data type. The function `whattype` returns the type of an expression while the function `type` can be used to check that an expression has a given type.

The function `op` can be used to find the components of a given type (i.e the expressions at the vertices which are joined to the root of the expression tree) while the function `nops` returns the number of such components.

Data structures More complicated programs involve manipulating and storing data. The representation of our data affects the algorithms that we write and how fast our programs will run. Maple offers a good set of data structures including lists (or vectors), sets, tables (or hash tables) and arrays.

Debugging facilities in Maple Writing software is bound to yield errors at some point. Since it is also possible to write large programs in Maple, there is a need of debugging facilities. Fortunately, Maple offers various methods:

The `mint` program This program is separate of the Maple system and used outside of Maple itself by a command such as `mint` $< file >$.

Mint will then produce a report on the code including likely errors. The amount of reported information can be controlled by a flag.

The `printlevel` facility Each statement which is executed in Maple has a certain level associated with it. Results of statements whose levels are no t higher than the global variable `printlevel` (default value is 2) are printed out during evaluation. Setting high values for `printlevel` leads to more information printed on the screen and can be very useful for debugging activities.

Tracing specific procedures Selective information can be obtained by using the built–in function `trace` or `debug` which can take one or more procedure names as arguments. This causes entry points (with argument values) and exit points (with result values) to be displayed as well as information on statements executed within the relevant procedures.

A.1.3 Building your own MAPLE package

Assume you have written a certain number of procedures concerning a specific topic and you would like to let others make use of your code. The most elegant way to do this is providing a package with on–line documentation and other advantages like the existing Maple packages. Unfortunately this is not (or just vaguely) described in the existing documentation.

So how do we proceed?

Suppose we have procedures `f1, f2, f3` and want to combine them in a package called `mypack` .

1. Create an empty archive in the directory `mylibdir` using the Maple Archive Manager: `march -c mylibdir 10` (the number 10 means that MARCH creates space for approximately 10 entries).

2. Create an index for the package: `mypack[f1]:='readlib( ‘mypack/f1‘ )':` `mypack[f2]:='readlib( ‘mypack/f2‘ )': mypack[f1]:='readlib( ‘mypack/f3‘ )':` .

   The functions `f1, f2, f3` are `readlib` defined such that Maple does not always load the whole package when we just want to use a single function of it. Suppose that this index code and internal functions (replace `internal_f` with `‘mypack/internal_f‘` for each internal function and simplify access to the internal functions using `macro (internal_f = ‘mypack/internal_f‘);`) that are used by different package functions are located in `mylibdir/src/mypack.mpl`. We will open a new Maple session, read in the code and save it in Maple's internal format with `> save ‘mylibdir/mypack.m‘;`.

   Note that it is necessary to restart Maple before and after doing this since `save ‘<filename>.m‘;` saves all defined names of the session in internal format. Using `save 'mypack','intf1','intf2',...,` `‘<filename>.m‘;` would avoid the restart but be rather cumbersome having many internal files.

3. Now we have to save the package procedures separately. We assume that the code of each package function `f_k` with corresponding internal functions that are not used by other package functions resides in `mylibdir/src/mypack_f_k.mpl` .Note that `f_k:=proc ... end:`has to be substituted with `‘mypack/f_k‘:=proc ... end:` according to the package table. Proceed analogously for all internal functions. Save the code for each package procedure : `>save ‘mylibdir/mypack/f_k.m‘;` and restart.

4. It remains to add the `.m` files to the archive: `march -a mylibdir mypack.m mypack` and `march -a mylibdir mypack/f_k.m mypack/f_k` for each package function.

5. The package is created but we have to tell Maple that there is a new directory to search for packages. We simply set `libname:=‘mydirlib‘,libname;` and add it to our `\$HOME/.mapleinit` file. When we start Maple from now on we can use our functions after `with(mypack);` or single functions after `with(mypack,\_k);` . It is often desirable that package functions can be individually accessed with the command > `mypack[f_k](args);` like in all packages supplied with Maple. To permit this capability we add `mypack:='readlib('mypack')':` to our `\$HOME/.mapleinit` file .

6. On–line documentation is indispensable even if no other persons will ever use the package. It is suggested to organize help pages like Maple does: FUNCTION, CALLING SEQUENCE, PARAMETERS, DESCRIPTION and last but not least EXAMPLES. Maple V R4 simplifies our task. We simply create a new worksheet for every package function, "construct" the corresponding help page (possibly using hyperlinks to refer to other functions) and save them in `mylibdir/mypack/help` as `.mws` files. Finally, we create our help database using Maple's `makehelp` facility:

   ```
   > readlib(makehelp):

   > makehelp(‘mypack‘,‘mylibdir/help/mypackhelp.mws‘,‘mylibdir‘):

   > makehelp(‘mypack,f1‘,‘mylibdir/help/f1help.mws‘,‘mylibdir‘):

   > ...
   ```

   Help on our package can now be obtained using `> ? mypack[f_k]`. Alternatively, we can use the Maple Save to Database option (in the Help menue). Maple asks us for a help topic (put in `mypack,f`), parent name (put in `mypack`), aliases (put in `f,mypack[f]`)and a directory of a help database (put in `mylibdir`) to save the current worksheet in as a Maple help file.

   Surprisingly `?f_k` does not work even after `with(mypack):` for both methods (due to a strange Maple policy).

This is the complete proceeding to install a package. If we want to change a package we have to save the modified source part as a `.m` file again and update the package with `march -u mylibdir <newsource> <newindex>`.

A.2 Why MAPLE ?

Since we are interested in determinant formulas of specially structured matrices of symbolic order with generally symbolic entries (such as polynomials) it is clear that we need a facility for symbolic mathematical computation.

This rules out the standard programming languages like C/C++ which is not disadvantageous because we don't have to deal with the issue of computation speed.

Thus we are left with computer algebra (CA) systems which all provide symbolic computation. There is quite a big number of CA systems available (like Scratchpad II/AXIOM, Mathematica, Maple, muMath/Derive, SAC/ALDES, Reduce etc.) with various large user areas (unmentioned various local CA at universities). Maple certainly is among the most well known CA systems. Its platform independence, ease of use and possibilities to extend the system are reasons for the popularity among different user communities.

Moreover, as we have seen, Maple offers all the needed symbolic computation methods as well as an easy programming language and the possibility to package our code nicely, which explains our choice.

All our code is developed using Maple V R4 . The code is divided in packages with on–line help and can be added to the Maple's share library. The packages including source code and help pages is available at:

`http://www.cs.uni-sb.de/users/mdenny/Bewohner/mark/Determinant.html`

Appendix B

Tutorial for the implemented Maple packages

This part contains the complete tutorial of the implemented packages. It is identical to the on–line documentation included in the packages. The help pages are organized like other Maple help pages.

B.1 The FRAMEFORMS package

This is the tutorial to the FRAMEFORMS package which is also available as on–line documentation.

Help For: Introduction to the FRAMEFORMS package

Calling Sequence:

function(args)
FRAMEFORMS[function](args)

Description:

- To use a FRAMEFORMS function, either define that function alone using the command with(FRAMEFORMS, function), or define all FRAMEFORMS functions using the command with(FRAMEFORMS). Alternatively, invoke the function using the long form FRAMEFORMS[function]. This long form notation is necessary whenever there is a conflict between a package function name and another function used in the same session.
- The functions available are: FrameformMatrix, GetBorderForm, Form7, Arrow, Nform, DBform, Rform, Frameform.
- This package provides a number of functions computing the determinant of matrices that have at most two rows and columns as well as at most two neighbouring central diagonals nonzero. It is possible to derive symbolic order determinant formulas for most of these matrices. Integer order evaluation is also possible.
- For more information on a particular function, see FRAMEFORMS[function].

See Also: **FRAMEFORMS, HESSENBERGandCONTINUANT, SYMMETRIC.**

Help For: Specification of a FRAMEFORMS matrix and the FrameformMatrix function

Description:

The only nonzero elements of a FRAMEFORMS matrix of order n may appear in at most two rows and columns, in the main diagonal, the 1st upper and lower sidediagonals or the corresponding counter diagonals. We allow at most two neighbouring diagonals to be nonzero. In case of two diagonals we require bordering nonzero rows and columns.

Specification:

The specification of such a matrix is a list of two element lists (each defining the location and a corresponding piecewise function in i):

- specL = [[location_1, fL_1] , ... , [location_k , fL_k]]
- location = row[k] | col[k] | diag[0] | diag[1] | diag[-1] | cdiag[0] | cdiag[1] | cdiag[-1] , 1 <= k <= n (k posint)
- fL = [[p_1..p_2, f_1] , [p_2+1..p_3, f_2] , ... , [p_k +1..p_(k+1), f_k]]

Restrictions:

- Interval bounds p_j integer or of the form n-q, q integer with 1 <= p_1 <= p_(k+1) <= n .
- Note: for diag[1],diag[-1],cdiag[1],cdiag[-1] we need p_(k+1)<=n-1).
- f_j function in i (otherwise constant) defined on the intervall [p_j..p_(k+1)] without roots in that interval.
- If p_1>1 or p_(j+1)<n the gaps are filled out with zeroes such that no clashes in the corners occur.This is only applicable in the case of standard frame form matrices,

Short Cuts:

- diag instead of diag[0] and cdiag instead of cdiag[0].
- p_j instead of p_j..p_j for one element intervals.
- f instead of [[1..n, f]].
- [p_1..p_2, f] instead of [[p_1..p_2, f]].

Function: FRAMEFORMS[FrameformMatrix] - frame form matrix

Calling Sequence:

FrameformMatrix(n,specL)

Parameters:

n	-	matrix order (as above)
specL	-	specification of the frame form matrix (as above)

Examples:

```
> with(FRAMEFORMS):
> FrameformMatrix(n,[ [row[1],a],[row[n],i] ,[cdiag,[[2..n-1,1]]] ]);
```

$$\begin{bmatrix} a & a & a & o & o & o & a \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & o & 0 & 0 & 0 \\ 0 & 0 & o & 0 & 0 & 0 & 0 \\ 0 & o & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 3 & o & o & o & n \end{bmatrix}$$

```
> FrameformMatrix(5,[ [row[1],[[1..2,x],[3..5,y]] ],[diag[-1],-1],[diag,[2..5,a]] ]);
```

$$\begin{bmatrix} x & x & y & y & y \\ -1 & a & 0 & 0 & 0 \\ 0 & -1 & a & 0 & 0 \\ 0 & 0 & -1 & a & 0 \\ 0 & 0 & 0 & -1 & a \end{bmatrix}$$

```
> FrameformMatrix(n+1, [ [row[1],a[i-1]],[cdiag,[2..n+1,x]],[cdiag[-1],-1]]);
```

$$\begin{bmatrix} a_0 & a_1 & a_2 & o & o & o & a_{n-1} & a_n \\ 0 & 0 & 0 & 0 & 0 & 0 & x & -1 \\ 0 & 0 & 0 & 0 & 0 & x & -1 & 0 \\ 0 & 0 & 0 & 0 & o & -1 & 0 & 0 \\ 0 & 0 & 0 & o & o & 0 & 0 & 0 \\ 0 & 0 & o & o & 0 & 0 & 0 & 0 \\ 0 & x & o & 0 & 0 & 0 & 0 & 0 \\ x & -1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

```
> FrameformMatrix(n,[[row[3],3],[diag,i],[col[n -1],[[1..3,3],[4..n,n-1]]],
[col[2],i]]);
```

$$\begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\
0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\
3 & 3 & 3 & 3 & 3 & o & o & o & 3 & 3 \\
0 & 4 & 0 & 4 & 0 & 0 & 0 & 0 & n-1 & 0 \\
0 & 5 & 0 & 0 & 5 & 0 & 0 & 0 & n-1 & 0 \\
0 & o & 0 & 0 & 0 & o & 0 & 0 & o & 0 \\
0 & o & 0 & 0 & 0 & 0 & o & 0 & o & 0 \\
0 & o & 0 & 0 & 0 & 0 & 0 & o & o & 0 \\
0 & n-1 & 0 & 0 & 0 & 0 & 0 & 0 & n-1 & 0 \\
0 & n & 0 & 0 & 0 & 0 & 0 & 0 & n-1 & n
\end{bmatrix}$$

```
>  FrameformMatrix(n,[ [col[n],[[1,x],[2..n-2,y]] ] ,[row[1],[[3..n-1,1+x]] ],
[diag[-1],[[1..5,q],[6..n-1,a[i]]] ] ]);
```

$$\begin{bmatrix}
0 & 0 & 1+x & 1+x & 1+x & 1+x & 1+x & o & o & o & 1+x & 1+x & x \\
q & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & y \\
0 & q & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & y \\
0 & 0 & q & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & y \\
0 & 0 & 0 & q & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & y \\
0 & 0 & 0 & 0 & q & 0 & 0 & 0 & 0 & 0 & 0 & 0 & y \\
0 & 0 & 0 & 0 & 0 & a_6 & 0 & 0 & 0 & 0 & 0 & 0 & y \\
0 & 0 & 0 & 0 & 0 & 0 & a_7 & 0 & 0 & 0 & 0 & 0 & o \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 & o \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & o \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & y \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{n-2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{n-1} & 0
\end{bmatrix}$$

See Also: GetBorderForm

Function: FRAMEFORMS[GetBorderForm] - transforms a general frame form matrix into border form such that the determinants are equivalent

Calling Sequence:

GetBorderForm(n, specL)
GetBorderForm(n, specL, print)

Parameters:

n	-	order of determinant (positive integer or symbolic expression of the form n+d,d integer)
specL	-	specification of the general frame form matrix (see FrameformMatrix for details)
print	-	optional directive

Description:

- The function GetBorderForm transformes the specified general frame form matrix into border from, i.e. nonzero rows or columns will be swapped to the matrix borders (updating the remaining matrix). The tranformation is only possible when we have only one nonzero diagonal, otherwise an error will be returned.
- A possible signfactor occuring during the transformation is multiplied to the first row of the resulting matrix.
- Note that the output matrix should not be used in further computations for symbolic orders, since Maple treats the abbreviating "dots" as normal matrix entries.
- The directive print (optional) prints the specified general frame form matrix before returning the transformed matrix.
- The command with(FRAMEFORMS) or with(FRAMEFORMS, GetBorderForm) allows the abbreviated form of this command after setting the libname appropriately.

Examples:

```
>  with(FRAMEFORMS):
>  GetBorderForm(n,[ [row[3],3] , [row[n-1],n-1] ,[diag,i] ],print);
```

$$Matrix :$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 3 & 3 & 3 & o & o & o & 3 & 3 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 \\ n-1 & n-1 & n-1 & n-1 & o & o & o & n-1 & n-1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n \end{bmatrix}$$

$$Transformed\ Matrix :$$

$$\begin{bmatrix} 3 & 3 & 3 & 3 & 3 & o & o & o & 3 & 3 & 3 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n & 0 \\ n-1 & n-1 & n-1 & n-1 & n-1 & o & o & o & n-1 & n-1 & n-1 \end{bmatrix}$$

```
>  GetBorderForm(n,[[row[3],3],[col[n-1],i],[dia g,i]],print);
```

$$Matrix:$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 3 & 3 & 3 & 3 & o & o & o & 3 & 3 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & o & 0 & 0 & o & 0 \\ 0 & 0 & 0 & 0 & 0 & o & 0 & o & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & o & o & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & n & n \end{bmatrix}$$

$$Transformed\ Matrix:$$

$$\begin{bmatrix} 3 & 3 & 3 & 3 & 3 & o & o & o & 3 & 3 & 3 \\ 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ o & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 & 0 \\ o & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 \\ o & 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 \\ n-2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-2 & 0 & 0 \\ n-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ n & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n \end{bmatrix}$$

```
> GetBorderForm(8,[ [row[2],[[1..1,d],[2..2,a],[3..3,b],[4..4,d],[5..5,c],[6..8,d]] ],
[row[5],[[1..2,e],[3..3,b],[4..4,e],[5..5,a],[6..8,e]] ],[col[3],[[1..2,b],[3..3,a],
[4..8,b]] ],[col[5],[[1..4,c],[5..5,a],[6..8,c]] ] , [diag,a] ] ,print);
```

$$\mathit{Matrix}:$$

$$\begin{bmatrix} a & 0 & b & 0 & c & 0 & 0 & 0 \\ d & a & b & d & c & d & d & d \\ 0 & 0 & a & 0 & c & 0 & 0 & 0 \\ 0 & 0 & b & a & c & 0 & 0 & 0 \\ e & e & b & e & a & e & e & e \\ 0 & 0 & b & 0 & c & a & 0 & 0 \\ 0 & 0 & b & 0 & c & 0 & a & 0 \\ 0 & 0 & b & 0 & c & 0 & 0 & a \end{bmatrix}$$

$$\mathit{Transformed\ Matrix}:$$

$$\begin{bmatrix} -b & -d & -a & -d & -d & -d & -d & -c \\ b & a & 0 & 0 & 0 & 0 & 0 & c \\ a & 0 & 0 & 0 & 0 & 0 & 0 & c \\ b & 0 & 0 & a & 0 & 0 & 0 & c \\ b & 0 & 0 & 0 & a & 0 & 0 & c \\ b & 0 & 0 & 0 & 0 & a & 0 & c \\ b & 0 & 0 & 0 & 0 & 0 & a & c \\ b & e & e & e & e & e & e & a \end{bmatrix}$$

See Also: FRAMEFORMS

Function: FRAMEFORMS[Form7] - determinant of an 7 - form matrix

Calling Sequence:

Form7(n, specL)
Form7(n, specL, print)
Form7(n, specL, check) or Form7(n, specL, check[k]) (k positive integer)
Form7(n, specL, print, check) or Form7(n, specL, print, check[k]) (k positive integer)

Parameters:

n	-	order of determinant (positive integer or symbolic expression of the form n+d,d integer)
specL	-	specification of the 7 form (see FrameformMatrix for details) A 7-form is a matrix with only one bordering row or column together with two appropriate neighbouring central diagonals nonzero.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function Form7 computes a formula for the determinant of the specified 7-form matrix. If necessary, the valid range for the formula is displayed.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified 7-form matrix before returning the determinant formula.
- The command with(FRAMEFORMS) or with(FRAMEFORMS, Form7) allows the abbreviated form of this command after setting the libname appropriately.

Examples:

```
> with(FRAMEFORMS):
> Form7(n+1,[ [row[1],a[i-1]],[diag,[2..n+1,x]],[diag[-1],-1] ],print,check);
```

Matrix :

$$\begin{bmatrix} a_0 & a_1 & a_2 & o & o & o & a_{n-1} & a_n \\ -1 & x & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & x & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & o & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & o & o & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & o & o & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & o & x & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & x \end{bmatrix}$$

Determinant :

$$n >= 4$$

$$\sum_{l\_=1}^{n+1} a_{l\_-1}\, x^{(n+1-l\_)}$$

```
> Form7(n,[ [col[1],[[2..3,a],[4..n-3,b],[n-2..n,c]]],[diag,1],[diag[1],-1]],
print,check);
```

Matrix :

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ b & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ b & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ o & 0 & 0 & 0 & 0 & o & o & 0 & 0 & 0 & 0 & 0 \\ o & 0 & 0 & 0 & 0 & 0 & o & o & 0 & 0 & 0 & 0 \\ o & 0 & 0 & 0 & 0 & 0 & 0 & o & o & 0 & 0 & 0 \\ b & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Determinant:$$

$$n >= 8$$

$$2\,a - 7\,b + 1 + b\,n + (-1)^{(2\,n)}\,b + 3\,(-1)^{(2\,n)}\,c$$

See Also: FRAMEFORMS

Function: FRAMEFORMS[Arrow] - determinant of an arrow shaped matrix

Calling Sequence:

Arrow(n, specL)
Arrow(n, specL, print)
Arrow(n, specL, check) or Arrow(n, specL, check[k])
Arrow(n, specL, print, check) or Arrow(n, specL, print, check[k])

Parameters:

n	-	order of determinant (positive integer or symbolic expression of the form n+d,d integer)
specL	-	specification of the arrow matrix (see FrameformMatrix for details) An arrow form is a matrix with only one nonzero row and adjacent column together with at most two appropriate neighbouring central diagonals nonzero.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function Arrow computes a formula for the determinant of the specified arrow shaped matrix. If necessary, the valid range for the formula is displayed.

- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of an integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified arrow shaped matrix before returning the determinant formula.
- The command with(FRAMEFORMS) or with(FRAMEFORMS, Arrow) allows the abbreviated form of this command.

Examples:

```
>  with(FRAMEFORMS):
>  Arrow(n,[ [row[1],a],[col[1],[2..n,b]],[diag,[2..n,1]] ],print,check);
```

$$\mathit{Matrix}\ :$$

$$\begin{bmatrix} a & a & a & o & o & o & a \\ b & 1 & 0 & 0 & 0 & 0 & 0 \\ b & 0 & 1 & 0 & 0 & 0 & 0 \\ o & 0 & 0 & o & 0 & 0 & 0 \\ o & 0 & 0 & 0 & o & 0 & 0 \\ o & 0 & 0 & 0 & 0 & o & 0 \\ b & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathit{Determinant}\ :$$

$$n\ >=\ 2$$

$$a - b\,a\,n + b\,a$$

```
>  Arrow(n,[ [row[n],a[i]],[col[1],[1..n-1,2]],[cdiag[-1],[1..n-2,i]] ],print,check);
```

$$\mathit{Matrix}\ :$$

$$\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ o & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ o & 0 & 0 & 0 & 0 & o & 0 & 0 \\ o & 0 & 0 & 0 & o & 0 & 0 & 0 \\ 2 & 0 & 0 & o & 0 & 0 & 0 & 0 \\ 2 & 0 & n-2 & 0 & 0 & 0 & 0 & 0 \\ a_1 & a_2 & o & o & o & a_{n-2} & a_{n-1} & a_n \end{bmatrix}$$

$$\mathit{Determinant}\ :$$

$$n\ >=\ 3$$

$$2\,(-1)^{(1+\mathrm{floor}(1/2\,n+1/2))}\,\Gamma(n-1)\,a_2$$

```
>  simplify( Arrow(n,[ [row[1],1],[col[n],[2..n,1]],[cdiag,[2..n,x]],[cdiag[-1],
[2..n-1,-1]] ],print,check[7]) ,assume=integer);
```

$$\textit{Matrix}:$$

$$\begin{bmatrix} 1 & 1 & 1 & o & o & o & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & x & 1 \\ 0 & 0 & 0 & 0 & 0 & x & -1 & 1 \\ 0 & 0 & 0 & 0 & o & -1 & 0 & o \\ 0 & 0 & 0 & o & o & 0 & 0 & o \\ 0 & 0 & o & o & 0 & 0 & 0 & o \\ 0 & x & o & 0 & 0 & 0 & 0 & 1 \\ x & -1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\textit{Determinant}:$$

$$n \;>=\; 3$$

$$-\frac{(-1)^{\mathrm{floor}(1/2\,n)}\,(-x^{(n+1)} + x^n - x^{(n-1)} + x^n\,n - x^{(n-1)}\,n + 1)}{(x-1)^2}$$

```
>  Arrow(n,[[row[3],3],[col[n-1],i],[diag,i]],print,check);
```

$$\textit{Matrix}:$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 3 & 3 & 3 & 3 & o & o & o & 3 & 3 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & o & 0 & 0 & o & 0 \\ 0 & 0 & 0 & 0 & 0 & o & 0 & o & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & o & o & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & n & n \end{bmatrix}$$

$$\textit{Determinant}:$$

$$n \;>=\; 7$$

$$\Gamma(n+1)$$

See Also: FRAMEFORMS

Function: FRAMEFORMS[Nform] - determinant of a N shaped matrix

Calling Sequence:

Nform(n, specL)
Nform(n, specL, print)

Nform(n, specL, check)
Nform(n, specL, print, check)

Parameters:

n	-	order of determinant (positive integer or symbolic expression of the form n+d,d integer)
specL	-	specification of the N shaped matrix (see FrameformMatrix for details) An N form is a matrix with only two rows or columns together with at most two appropriate neighbouring central diagonals nonzero.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function Nform computes a formula for the determinant of the given N shaped matrix. If necessary, the valid range for the formula is displayed.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of an integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified N shaped matrix before returning the determinant formula.
- The command with(FRAMEFORMS, Nform) or with(FRAMEFORMS) allows the abbreviated form of this command after setting the libname appropriately.

Examples:

```
> with(FRAMEFORMS):
> Nform(n,[ [col[1],a[i]],[col[n],b[n-i]],[diag,[2..n-1,i]] ],print,check);
```

Matrix :

$$\begin{bmatrix} a_1 & 0 & 0 & 0 & 0 & 0 & 0 & b_{n-1} \\ a_2 & 2 & 0 & 0 & 0 & 0 & 0 & b_{n-2} \\ a_3 & 0 & 3 & 0 & 0 & 0 & 0 & b_{n-3} \\ o & 0 & 0 & o & 0 & 0 & 0 & o \\ o & 0 & 0 & 0 & o & 0 & 0 & o \\ o & 0 & 0 & 0 & 0 & o & 0 & o \\ a_{n-1} & 0 & 0 & 0 & 0 & 0 & n-1 & b_1 \\ a_n & 0 & 0 & 0 & 0 & 0 & 0 & b_0 \end{bmatrix}$$

Determinant :

$$n >= 3$$

$$-(-a_1\, b_0 + b_{n-1}\, a_n)\, \Gamma(n)$$

```
>  simplify( Nform(n,[ [row[1],1],[row[n],i],[diag,[2..n-1,1]],[diag[-1],[1..n-2,-1]] ],
print,check) ,assume=integer);
```

$$Matrix:$$

$$\begin{bmatrix} 1 & 1 & o & o & o & 1 & 1 & 1 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & o & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & o & o & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & o & o & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & o & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 1 & 2 & o & o & o & n-2 & n-1 & n \end{bmatrix}$$

$$Determinant:$$

$$n >= 3$$

$$\frac{1}{2}n^2 - \frac{1}{2}n$$

```
>  Nform(n,[[row[3],a],[row[n-1],x[i]],[diag,[[1 ..3,a],[4..n,x[i]]]]],print,check);
```

$$Matrix:$$

$$\begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a & a & a & a & a & o & o & o & a & a \\ 0 & 0 & 0 & x_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 \\ x_1 & x_2 & x_3 & x_4 & x_5 & o & o & o & x_{n-1} & x_n \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_n \end{bmatrix}$$

$$Determinant:$$

$$n >= 7$$

$$(a\,x_{n-1} - x_3\,a)\,a^2\,(\prod_{l\_=3}^{-3+n} x_{l\_+1})\,x_n$$

See Also: FRAMEFORMS

Function: FRAMEFORMS[Rform] - determinant of an R shaped matrix

Calling Sequence:

Rform(n, specL)
Rform(n, specL, print)
Rform(n, specL, check) or Rform(n, specL, check[k]) (k positive integer)
Rform(n, specL, print, check) or Rform(n, specL, print, check[k]) (k positive integer)

Parameters:

n	-	order of determinant (positive integer or symbolic expression of the form n+d,d integer)
specL	-	specification of the R shaped matrix (see FrameformMatrix for details) An R form is a matrix with two nonzero columns and one nonzero row (or vice versa) together with at most two appropriate neighbouring central diagonals nonzero.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function Rform computes a formula for the determinant of the given R shaped matrix. If necessary, the valid range for the formula is displayed.
- Unfortunately, the function doesn't always work correctly in the case of symbolic order and two diagonals if the diagonal generating functions are functions in "i" (it works however, if "i" only appears as index like in f[i]). This is probably due to Maple's attempt finding closed forms for complicated sums and products.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified R shaped matrix before returning the determinant formula.
- The command with(FRAMEFORMS, Rform) or with(FRAMEFORMS) allows the abbreviated form of this command after setting the libname appropriately.

Examples:

```
> with(FRAMEFORMS):
> Rform(n,[ [row[1],1],[col[1],[2..n,a]],[col[n],[2..n,b[i]]],[diag,[2..n-1,c]] ]
,print,check);
```

Matrix :

$$\begin{bmatrix} 1 & 1 & 1 & o & o & o & 1 & 1 \\ a & c & 0 & 0 & 0 & 0 & 0 & b_2 \\ a & 0 & c & 0 & 0 & 0 & 0 & b_3 \\ o & 0 & 0 & o & 0 & 0 & 0 & o \\ o & 0 & 0 & 0 & o & 0 & 0 & o \\ o & 0 & 0 & 0 & 0 & o & 0 & o \\ a & 0 & 0 & 0 & 0 & 0 & c & b_{n-1} \\ a & 0 & 0 & 0 & 0 & 0 & 0 & b_n \end{bmatrix}$$

Determinant :

$n \;>=\; 4$

$$b_n\,(c^{(n-2)} - a\,c^{(-3+n)}\,n + 2\,a\,c^{(-3+n)}) - a\,(c^{(n-2)} - c^{(-3+n)}\,(\sum_{l\_=2}^{n-1} b_{l\_}))$$

```
>  Rform(n,[ [col[1],a[i]],[row[1],[2..n,b[i]]],[row[n],[2..n,1]],[diag[-1],
[2..n-2,i]] ],print,check[9]);
```

Matrix :

$$\begin{bmatrix} a_1 & b_2 & b_3 & o & o & o & b_{n-2} & b_{n-1} & b_n \\ a_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_3 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ o & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ o & 0 & 0 & o & 0 & 0 & 0 & 0 & 0 \\ o & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 \\ a_{n-2} & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 \\ a_{n-1} & 0 & 0 & 0 & 0 & 0 & n-2 & 0 & 0 \\ a_n & 1 & 1 & o & o & o & 1 & 1 & 1 \end{bmatrix}$$

Determinant :

$n \;>=\; 5$

$$(-1)^n\,(b_{n-1} - b_n)\,(\prod_{l\_=1}^{-3+n} (n - l\_ - 1))\,a_2$$

```
>  Rform(n,[[row[1],1],[col[1],i],[col[n-1],1],[ cdiag[1],i]],print,check);
```

Matrix :

$$\begin{bmatrix} 1 & 1 & o & o & o & 1 & 1 \\ 2 & 0 & 0 & 0 & 2 & 1 & 0 \\ o & 0 & 0 & o & 0 & o & 0 \\ o & 0 & o & 0 & 0 & o & 0 \\ o & o & 0 & 0 & 0 & o & 0 \\ n-1 & 0 & 0 & 0 & 0 & 1 & 0 \\ n & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$Determinant:$$

$$n >= 4$$

$$-(-1)^{\mathrm{floor}(1/2\,n)}\,(-1)^n\,\Big(\prod_{l\_=1}^{-3+n}(n-l\_-1)\Big)$$

See Also: **FRAMEFORMS, FRAMEFORMS[Arrow]**

Function: FRAMEFORMS[DBform] - determinant of a DB-matrix

Calling Sequence:

DBform(n, specL)
DBform(n, specL, print)
DBform(n, specL, check) or DBform(n, specL, check[k]) (k positive integer)
DBform(n, specL, print, check) or DBform(n, specL, print, check[k]) (k positive integer)

Parameters:

n	-	order of determinant (positive integer or symbolic expression of the form n+d,d integer)
specL	-	specification of the DB form (see FrameformMatrix for details) An DB form is a matrix with only the bordering rows and columns together with a diagonal non-zero.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function DBform computes a formula for the determinant of the given DB shaped matrix. If necessary, the valid range for the formula is displayed.

  No formula can be obtained in the case of two neighbouring diagonals or if both rows and columns don't contain a constant part of symbolic length (i.e. dependent on n).

- The directive print (optional) prints the specified DB shaped matrix before returning the determinant formula.

- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a finite order matrix is computed using the standart det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.

- The command with(FRAMEFORMS, DBform) or with(FRAMEFORMS) allows the abbreviated form of this command after setting libname appropriately.

Examples:

```
> with(FRAMEFORMS):

> DBform(n,[ [row[1],a],[row[n],b[i]],[col[1],[2..n-1,1]],[col[n],[2..n-1,x[i]]],
[diag[-1],[2..n-2,3]] ],print,check[9]);
```

Matrix :

$$\begin{bmatrix} a & a & a & o & o & o & a & a & a \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_2 \\ 1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & x_3 \\ o & 0 & 3 & 0 & 0 & 0 & 0 & 0 & o \\ o & 0 & 0 & o & 0 & 0 & 0 & 0 & o \\ o & 0 & 0 & 0 & o & 0 & 0 & 0 & o \\ 1 & 0 & 0 & 0 & 0 & o & 0 & 0 & x_{n-2} \\ 1 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & x_{n-1} \\ b_1 & b_2 & b_3 & o & o & o & b_{n-2} & b_{n-1} & b_n \end{bmatrix}$$

Determinant :

$$n >= 5$$

$$(-1)^{(n+2)}\, x_2\, (b_{n-1}\, (\frac{2}{27}\, 3^n\, a - \frac{1}{81}\, 3^n\, a\, n) - a\, (\frac{1}{27}\, 3^n\, b_1 - \frac{1}{81}\, 3^n\, (\sum_{l\_=2}^{n-2} b_{l\_})))$$
$$-\, (-1)^{(n-1)}\, (a\, (\frac{1}{27}\, 3^n\, b_n - \frac{1}{81}\, 3^n\, (\sum_{l\_=2}^{n-2} b_{n-l\_}\, x_{n+1-l\_})) - b_{n-1}\, (\frac{1}{27}\, 3^n\, a - \frac{1}{81}\, 3^n\, a\, (\sum_{l\_=2}^{n-2} x_{n+1-l\_})))$$

```
> DBform(n,[ [row[1],i],[row[n],a],[col[1],[[2,z],[3..n-2,1],[n-1,y]]],[col[n],
[2..n-1,x]],[diag,[2..n-1,k]] ],print,check[9]);
```

Matrix :

$$\left[\begin{array}{cccccccccc}
1 & 2 & 3 & 4 & o & o & o & n-2 & n-1 & n \\
z & k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x \\
1 & 0 & k & 0 & 0 & 0 & 0 & 0 & 0 & x \\
1 & 0 & 0 & k & 0 & 0 & 0 & 0 & 0 & x \\
o & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 & o \\
o & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & o \\
o & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & o \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & k & 0 & x \\
y & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k & x \\
a & a & a & a & o & o & o & a & a & a
\end{array}\right]$$

Determinant :

$n >= 6$

$$\left(-\frac{n}{x}+1-2\,\frac{-1+z}{k}-\frac{(n-1)\,(-1+y)}{k}\right)\left(k^{(n-2)}\,a-a\,x\,k^{(-3+n)}\,n+2\,a\,x\,k^{(-3+n)}\right)$$
$$-\left(-\frac{a}{x}+a-\frac{a\,(-1+z)}{k}-\frac{a\,(-1+y)}{k}\right)\left(k^{(n-2)}\,n-\frac{1}{2}\,x\,k^{(-3+n)}\,n^2+\frac{1}{2}\,x\,k^{(-3+n)}\,n+x\,k^{(-3+n)}\right)$$

```
>  DBform(d+2,[[col[1],1],[col[d+2],[[1,s[1]^ 2], [2..d+1,t[i-1]^ 2],[d+2,d*t^ 2]]],
[diag,[[2..d+1,t[i-1]]]],[row[1],[2,s[1]]],[row[d+2],[2..d+1,t]]],print,check[7]);
```

Matrix :

$$\left[\begin{array}{ccccccccc}
1 & s_1 & 0 & 0 & o & o & o & 0 & {s_1}^2 \\
1 & t_1 & 0 & 0 & 0 & 0 & 0 & 0 & {t_1}^2 \\
1 & 0 & t_2 & 0 & 0 & 0 & 0 & 0 & {t_2}^2 \\
1 & 0 & 0 & t_3 & 0 & 0 & 0 & 0 & {t_3}^2 \\
o & 0 & 0 & 0 & o & 0 & 0 & 0 & o \\
o & 0 & 0 & 0 & 0 & o & 0 & 0 & o \\
o & 0 & 0 & 0 & 0 & 0 & o & 0 & o \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & t_d & {t_d}^2 \\
1 & t & t & t & o & o & o & t & d\,t^2
\end{array}\right]$$

Determinant :

$n >= 5$

$$\left(1-\frac{s_1}{t_1}\right)\left(\left(\prod_{l\_=1}^{d} t_{d+1-l\_}\right)d\,t^2-t\left(\prod_{l\_=1}^{d} t_{d+1-l\_}\right)\left(\sum_{l\_=2}^{d+1} t_{d+2-l\_}\right)\right)$$
$$-\left({s_1}^2-t_1\,s_1\right)\left(\left(\prod_{l\_=1}^{d} t_{d+1-l\_}\right)-t\left(\prod_{l\_=1}^{d} t_{d+1-l\_}\right)\left(\sum_{l\_=2}^{d+1}\frac{1}{t_{d+2-l\_}}\right)\right)$$

```
>  DBform(d+2,[[col[1],1],[col[d+2],[[1,d*t^ 2],[ 2..d+1,t[i-1]2̂],[d+2,d*s^ 2]]],[diag,
[2..d+1,t[i-1]]],[row[1],[2..d+1,t]],[row[d+2],[2..d+1,s]]],print,check);
```

Matrix :

$$\begin{bmatrix} 1 & t & t & o & o & o & t & d\,t^2 \\ 1 & t_1 & 0 & 0 & 0 & 0 & 0 & {t_1}^2 \\ 1 & 0 & t_2 & 0 & 0 & 0 & 0 & {t_2}^2 \\ o & 0 & 0 & o & 0 & 0 & 0 & o \\ o & 0 & 0 & 0 & o & 0 & 0 & o \\ o & 0 & 0 & 0 & 0 & o & 0 & o \\ 1 & 0 & 0 & 0 & 0 & 0 & t_d & {t_d}^2 \\ 1 & s & s & o & o & o & s & d\,s^2 \end{bmatrix}$$

Determinant :

$n\ >=\ 4$

$$(-\frac{t}{s}+1)\,((\prod_{l\_=1}^{d} t_{d+1-l\_})\,d\,s^2 - s\,(\prod_{l\_=1}^{d} t_{d+1-l\_})\,(\sum_{l\_=2}^{d+1} t_{d+2-l\_}))$$
$$-\,(-d\,s\,t+d\,t^2)\left((\prod_{l\_=1}^{d} t_{d+1-l\_}) - s\,(\prod_{l\_=1}^{d} t_{d+1-l\_})\left(\sum_{l\_=2}^{d+1}\frac{1}{t_{d+2-l\_}}\right)\right)$$

```
>  DBform(n,[ [row[2],[[1..1,d],[2..2,a],[3..3,b],[4..4,d],[5..5,c],[6..n,d]] ],[row[5],
[[1..2,e],[3..3,b],[4..4,e],[5..5,a],[6..n,e]] ],[col[3],[[1..2,b],[3..3,a],[4..n,b]] ],
[col[5], [[1..4,c],[5..5,a],[6..n,c]] ] , [diag,a] ],print,check);
```

Matrix :

$$\begin{bmatrix} a & 0 & b & 0 & c & 0 & 0 & 0 & 0 & 0 & 0 \\ d & a & b & d & c & d & d & o & o & o & d \\ 0 & 0 & a & 0 & c & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & b & a & c & 0 & 0 & 0 & 0 & 0 & 0 \\ e & e & b & e & a & e & e & o & o & o & e \\ 0 & 0 & b & 0 & c & a & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & b & 0 & c & 0 & a & 0 & 0 & 0 & 0 \\ 0 & 0 & o & 0 & o & 0 & 0 & o & 0 & 0 & 0 \\ 0 & 0 & o & 0 & o & 0 & 0 & 0 & o & 0 & 0 \\ 0 & 0 & o & 0 & o & 0 & 0 & 0 & 0 & o & 0 \\ 0 & 0 & b & 0 & c & 0 & 0 & 0 & 0 & 0 & a \end{bmatrix}$$

Determinant :

$n\ >=\ 6$

$$(-b+a)\,(a^{(n-1)} + c\,e\,a^{(n-4)}\,n\,d - c\,e\,a^{(-3+n)}\,n - 3\,c\,e\,a^{(n-4)}\,d + 2\,a^{(-3+n)}\,c\,e) + a^{(n-2)}\,b\,(a-c)$$

See Also: **FRAMEFORMS, FRAMEFORMS[Arrow], FRAMEFORMS[Rform]**

Function: FRAMEFORMS[Frameform] - determinant of an arbitrary frame form matrix

Calling Sequence:

Frameform(n, specL)
Frameform(n, specL, print)
Frameform(n, specL, check) or Frameform(n, specL, check[k])
Frameform(n, specL, print, check) or Frameform(n, specL, print, check[k])

Parameters:

n	-	order of determinant (positive integer or symbolic expression of the form n+d,d integer)
specL	-	specification of the frame form matrix (see FrameformMatrix for details)
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function Frameform computes a formula for the determinant of the specified frame form matrix. If necessary, the valid range for the formula is displayed. This is the most general function in this package; it determines the type of the frame form (e.g. arrow type or form7 type, etc.) and calls the appropriate function.

- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.

- The directive print (optional) prints the specified arrow shaped matrix before returning the determinant formula.

- The command with(FRAMEFORMS) or with(FRAMEFORMS, Frameform) allows the abbreviated form of this command after setting the libname appropriately.

Examples:

```
> with(FRAMEFORMS):
> Frameform(n,[ [cdiag[1],1],[col[n],2],[row[n],[1..n-1,b]] ],print,check);
```

Matrix :

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & o & 0 & 0 & o \\ 0 & 0 & o & 0 & 0 & 0 & o \\ 0 & o & 0 & 0 & 0 & 0 & o \\ 1 & 0 & 0 & 0 & 0 & 0 & 2 \\ b & b & o & o & o & b & 2 \end{bmatrix}$$

Determinant :

$n \;>=\; 3$

$$2\,(-1)^{(n+\mathrm{floor}(1/2\,n))}\,(-1+b\,n-b)$$

```
>  Frameform(n,[ [diag,1],[col[1],[2..n,i]],[col[n],[1..n-1,x^2]] ],print,check[9]);
```

Matrix :

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & x^2 \\ 2 & 1 & 0 & 0 & 0 & 0 & x^2 \\ o & 0 & o & 0 & 0 & 0 & o \\ o & 0 & 0 & o & 0 & 0 & o \\ o & 0 & 0 & 0 & o & 0 & o \\ n-1 & 0 & 0 & 0 & 0 & 1 & x^2 \\ n & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Determinant :

$n \;>=\; 3$

$$1 - x^2\,n$$

```
>  Frameform(n,[[row[3],3],[col[n-2],[[1..3,3],[ 4..n,n-2]]],[diag,i]],print,check);
```

Matrix :

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 3 & 3 & 3 & 3 & 3 & o & o & o & 3 & 3 & 3 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & n-2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & n-2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & o & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & o & 0 & o & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & o & o & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-2 & n-1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-2 & 0 & n \end{bmatrix}$$

$$Determinant:$$

$$n\ >=\ 8$$

$$\Gamma(n+1)$$

See Also: FRAMEFORMS

B.2 The ALTERNANT package

This is the tutorial to the ALTERNANT package which is also available as on–line documentation.

Help For: Introduction to the ALTERNANT package

Calling Sequence:

function(args)
ALTERNANT[function](args)

Description:

- To use an ALTERNANT function, either define that function alone using the command with(ALTERNANT, function), or define all ALTERNANT functions using the command with(ALTERNANT). Alternatively, invoke the function using the long form ALTERNANT[function]. This long form notation is necessary whenever there is a conflict between a package function name and another function used in the same session.
- The functions available are: Alternant, AlternantMatrix, DP, CSF, evalcsf, CSFcofactorMatrix, ESF, ESFcofactorMatrix, S_to_esf, evalesf, DoubleAlternant, DoubleAlternantMatrix, DoubleAlternantCofactorMatrix, TrigAlternant
- This package provides a number of functions computing the determinant of an alternant (a matrix with columns generated by different functions (polynomials) in variables x[i]). Different methods can be used to obtain results with different representation of the symmetric sum which is cofactor of the difference product in every alternant.

  Unfortunately a formula for an alternant of symbolic order n can only be determined for special alternants:

  e.g. alternants with maximum degree of n+k (k integer) for the generating polynomial(s).
- For more information on a particular function, see ?ALTERNANT[function].

See Also: **FRAMEFORMS, HESSENBERGandCONTINUANT, SYMMETRIC**

Function: ALTERNANT[AlternantMatrix] - matrix of the specified alternant

Calling Sequence:

AlternantMatrix(n,x,F)

Parameters:

n	-	alternant order (symbolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
F	-	list of piecewise function specifications specifiying the alternant (see the Specification for details)
method	-	
print	-	optional display directive

Specification:

The piecewise specification with column generating functions should look like this:

[[1..p1,f_1], [p1+1..p2,f_2], ... , [p_k+1..n,f_k]]

with 1 <= p1 < p2 < ... < p_k <= n.

The positions p have to be integers, in the symbolic case of the form d or n-d, with d posint.

The functions f in each interval have to be polynomials in x[i] of the form

$\sum_{l=1}^{k} (c_l \, {x_i}^{q_l} + d_l \, {x_i}^{(a\,j+p_l)})$

with c_l, d_l integer or variables distinct from i; q_l, p_l integer and k,a posint.

That is, a sublist L=[p1..p2,x[i]^j +1] in the piecewise function specification list means, that the column p1 is generated by the function x[i]^p1 +1 and so on until the column p2 which is generated by the function x[i]^p2 +1.

Short Cuts:

- If the matrix is generated by a single generating function f, we may write f instead of [[1..n,f]].
- If an interval contains only one element, we may write [p,f] instead of [p..p,f].

Description:

- The function AlternantMatrix returns the matrix of the specified alternant.
- Note, that for symbolic alternant order, the returned matrix should be used for illustrative purposes only, since the dots "o" used to abbreviate the symbolic case are treated as usual matrix entries by Maple in any further computation.
- This function is part of the ALTERNANT package and so can be used in the form AlternantMatrix(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,AlternantMatrix).

Examples:

```
>  with(ALTERNANT):
>  AlternantMatrix(4,x,x[i]^j);
```

$$\begin{bmatrix} x_1 & {x_1}^2 & {x_1}^3 & {x_1}^4 \\ x_2 & {x_2}^2 & {x_2}^3 & {x_2}^4 \\ x_3 & {x_3}^2 & {x_3}^3 & {x_3}^4 \\ x_4 & {x_4}^2 & {x_4}^3 & {x_4}^4 \end{bmatrix}$$

```
> AlternantMatrix(n,x,[[1..2,x[i]^(j-1)],[3..n- 1,x[i]^j],[n..n,4*x[i]^2]]);
```

$$\begin{bmatrix} 1 & x_1 & {x_1}^3 & {x_1}^4 & o & o & {x_1}^{(n-2)} & {x_1}^{(n-1)} & 4\,{x_1}^2 \\ 1 & x_2 & {x_2}^3 & {x_2}^4 & o & o & {x_2}^{(n-2)} & {x_2}^{(n-1)} & 4\,{x_2}^2 \\ 1 & x_3 & {x_3}^3 & {x_3}^4 & o & o & {x_3}^{(n-2)} & {x_3}^{(n-1)} & 4\,{x_3}^2 \\ 1 & x_4 & {x_4}^3 & {x_4}^4 & o & o & {x_4}^{(n-2)} & {x_4}^{(n-1)} & 4\,{x_4}^2 \\ o & o & o & o & o & o & o & o & o \\ o & o & o & o & o & o & o & o & o \\ 1 & x_{n-2} & {x_{n-2}}^3 & {x_{n-2}}^4 & o & o & {x_{n-2}}^{(n-2)} & {x_{n-2}}^{(n-1)} & 4\,{x_{n-2}}^2 \\ 1 & x_{n-1} & {x_{n-1}}^3 & {x_{n-1}}^4 & o & o & {x_{n-1}}^{(n-2)} & {x_{n-1}}^{(n-1)} & 4\,{x_{n-1}}^2 \\ 1 & x_n & {x_n}^3 & {x_n}^4 & o & o & {x_n}^{(n-2)} & {x_n}^{(n-1)} & 4\,{x_n}^2 \end{bmatrix}$$

```
> AlternantMatrix(5,x,[[1..3,1+x[i]^j],[4..5,a- b*x[i]^(j+1)+x[i]]]);
```

$$\begin{bmatrix} 1+x_1 & 1+{x_1}^2 & 1+{x_1}^3 & a-b\,{x_1}^5+x_1 & a-b\,{x_1}^6+x_1 \\ 1+x_2 & 1+{x_2}^2 & 1+{x_2}^3 & a-b\,{x_2}^5+x_2 & a-b\,{x_2}^6+x_2 \\ 1+x_3 & 1+{x_3}^2 & 1+{x_3}^3 & a-b\,{x_3}^5+x_3 & a-b\,{x_3}^6+x_3 \\ 1+x_4 & 1+{x_4}^2 & 1+{x_4}^3 & a-b\,{x_4}^5+x_4 & a-b\,{x_4}^6+x_4 \\ 1+x_5 & 1+{x_5}^2 & 1+{x_5}^3 & a-b\,{x_5}^5+x_5 & a-b\,{x_5}^6+x_5 \end{bmatrix}$$

See Also: ALTERNANT[Alternant]

Function: ALTERNANT[Alternant] - determinant of a specified alternant

Calling Sequence:

Alternant(n,x,F)
Alternant(n,x,F,method)
Alternant(n,x,F,print)
Alternant(n,x,F,check)
Alternant(n,x,F,method,print)
Alternant(n,x,F,method,check)
Alternant(n,x,F,method,print,check)

Parameters:

n	-	alternant order (symbolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
F	-	list of piecewise function specifications specifiying the alternant (see the Specification for details)
method	-	optional computation directive: esf, csf or normal
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function Alternant tries to compute the determinant formula of the specified alternant. If necessary, the valid range of the formula is displayed..
- If the optional directive esf is given, it is tried to compute a elementary symmetric function representation of the determinant formula. For symbolic alternant order, this is only possible for the following types of alternants: The Alternant has maximum degree of n+d, d integer and the specification list F is of the form: Either the sublist [p..n-q,f] specifying the infinite intervall, we require f to be a monomial or we have the form $[[1..n,\, c_1\, {x_i}^{(j+k_1)} + c_2\, {x_i}^{(j+k_2)}]]$, c1,c2,k1,k2 integer or the form $\left[\left[1..n,\, c\, {x_i}^{(j+d)} + \left(\sum_{l=1}^{k} c_l\, {x_i}^{q_l}\right)\right]\right]$ with c,d,k,c_l,q_l integer.
- If the optional directive csf is given, it is tried to compute a complete symmetric function representation of the determinant formula. For symbolic alternant order, this is only possible for the following type of function specification: $[1..n-p,\, c\, {x_i}^{(j+k)}]$ has to be the first sublist, followed by arbitrary other sublists.
- If the optional directive normal is given, it is tried to compute a formula via clever factoring or simulation of column operations. We require that the definition is complete, not piecewise, i.e. of the form [[1..n,f]]. The maximum degree of f should be n+k, k integer. For this method only, we also allow generating functions of the form $\mathrm{q}(x_i)\, \mathrm{p}(x_i)^{\mathrm{l}(j)}$ with q(x[i]),p(x[i]) polynomials of finite degree independent of j and $\mathrm{l}(j) = k\, j + d$ k,d integer.
- If no directive concerning the computation method is given, it is tried to use the most appropriate for each special input.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a finite order matrix is computed using the standart det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified alternant matrix before returning the determinant formula.
- This function is part of the ALTERNANT package and so can be used in the form Alternant(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,Alternant).

Examples:

```
> with(ALTERNANT):
> Alternant(n,x,[[1..n-2,3*x[i]^(j-2)],[n-1..n, 2+a*x[i]^j]],esf,check);
```

$$\textit{formula valid for },\ 2 < n$$

$$\frac{1}{3}\left(\prod_{k\_=1}^{n} \frac{1}{x_{k\_}}\right) 3^{(n-1)}\, a^2\, (\mathrm{S}(2,\, n,\, x)^2 - \mathrm{S}(1,\, n,\, x)\, \mathrm{S}(3,\, n,\, x))\, \mathit{ALTERNANT}_{DP}(n,\, x,\, x_i)$$

```
> Alternant(n,x,x[i]^(j-1)+x[i]^j,esf,check);
```

$$((-1)^n\, \mathrm{S}(n,\, n,\, x) - (-1)^n\, (\sum_{l\_=0}^{n-1} \mathrm{S}(n-1-l\_,\, n,\, x)\, (-1)^{l\_}))\, \mathit{ALTERNANT}_{DP}(n,\, x,\, x_i)$$

```
>  Alternant(n,x, 2+x[i]^j+a*x[i]^3,esf,check);
```

$$(-1)^n\,(\mathrm{S}(n,\,n,\,x)+a\,\mathrm{S}(n,\,n,\,x)-2\,(\sum_{l\_=0}^{n-1}\mathrm{S}(l\_,\,n,\,x)))\,\mathit{ALTERNANT}_{DP}(n,\,x,\,x_i)$$

```
>  Alternant(n,x,1+x[i]^3+x[i]^j,normal,check);
```

$$\sum_{l\_=1}^{n}(-1)^{(l\_+1)}\,(1+{x_{l\_}}^3+x_{l\_})\,(\prod_{k\_=1}^{l\_-1}x_{k\_})\,(\prod_{k\_=l\_+1}^{n}x_{k\_})\,(\prod_{k\_=1}^{l\_-1}(x_{k\_}-1))\,(\prod_{k\_=l\_+1}^{n}(x_{k\_}-1))$$
$$\left(\prod_{i\_=1}^{l\_-1}(\prod_{j\_=i\_+1}^{l\_-1}(x_{j\_}-x_{i\_}))\,(\prod_{j\_=l\_+1}^{n}(x_{j\_}-x_{i\_}))\right)\,(\prod_{i\_=l\_+1}^{n}(\prod_{j\_=i\_+1}^{n}(x_{j\_}-x_{i\_})))$$

```
>  Alternant(n,x,[[1..n-1,2*x[i]^(j)],[n..n,a*x[ i]^n-x[i]^(n+1)]],csf,check);
```

$$\mathit{formula\ valid\ for}\ ,\,1<n$$

$$(-2^{(n-1)}\,(\prod_{l\_=1}^{n}x_{l\_})\,\mathrm{csf}(1,\,n,\,x)+2^{(n-1)}\,a\,(\prod_{l\_=1}^{n}x_{l\_}))\,\mathit{ALTERNANT}_{DP}(n,\,x,\,x_i)$$

```
>  Alternant(n,x,(x[i]+2)*(a+b*x[i]+c*x[i]^2)^j, normal);
```

$$\left(\prod_{k\_=1}^{n}(x_{k\_}+2)\,(a+b\,x_{k\_}+c\,{x_{k\_}}^2)\right)\,\mathit{ALTERNANT}_{DP}(n,\,x,\,a+b\,x_i+c\,{x_i}^2)$$

```
>  Alternant(n+2,y, l*y[i]^j,print,check);
```

$$\mathit{Matrix}\ :$$

$$\begin{bmatrix} l\,y_1 & l\,{y_1}^2 & o & o & l\,{y_1}^{(n+1)} & l\,{y_1}^{(n+2)} \\ l\,y_2 & l\,{y_2}^2 & o & o & l\,{y_2}^{(n+1)} & l\,{y_2}^{(n+2)} \\ o & o & o & o & o & o \\ o & o & o & o & o & o \\ l\,y_{n+1} & l\,{y_{n+1}}^2 & o & o & l\,{y_{n+1}}^{(n+1)} & l\,{y_{n+1}}^{(n+2)} \\ l\,y_{n+2} & l\,{y_{n+2}}^2 & o & o & l\,{y_{n+2}}^{(n+1)} & l\,{y_{n+2}}^{(n+2)} \end{bmatrix}$$

$$\mathit{Determinant}\ :$$

$$\frac{l^{(n+3)}\,(\prod_{k\_=1}^{n+2}y_{k\_})\,\mathit{ALTERNANT}_{DP}(n+2,\,y,\,y_i)}{l}$$

See Also: **ALTERNANT[AlternantMatrix], ALTERNANT[ESFcofactorMatrix], ALTERNANT[CSFcofactorMatrix], ALTERNANT[DP]**

Function: ALTERNANT[DP] - computes the difference product of a given function

Calling Sequence:

DP(n,x,f)
DP(n,x,f)

Parameters:

n	-	order of the difference product (symolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
f	-	polynomial in x[i]

Description:

- The function DP computes the difference product prod_{i<j} (f(x_j)-f(x_i)).
- If an integer value for n is given, DP evaluates the difference product of order n. Otherwise DP returns the unevaluated difference product of symbolic order n.
- This function is part of the ALTERNANT package and so can be used in the form DP(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,DP).

Examples:

```
> with(ALTERNANT):
> DP(n,x,x[i]);
```

$$\prod_{i_1=1}^{n}\left(\prod_{i_2=i_1+1}^{n}(x_{i_2}-x_{i_1})\right)$$

```
> DP(4,x,x[i]);
```

$$(x_2-x_1)(x_3-x_1)(x_4-x_1)(x_3-x_2)(x_4-x_2)(x_4-x_3)$$

```
> DP(n,x,2*x[i]^2-x[i]+3);
```

$$\prod_{i_1=1}^{n}\left(\prod_{i_2=i_1+1}^{n}(2\,{x_{i_2}}^2-x_{i_2}-2\,{x_{i_1}}^2+x_{i_1})\right)$$

See Also: **ALTERNANT[Alternant], ALTERNANT[DoubleAlternant]**

Function: ALTERNANT[CSF] - complete symmetric function

Calling Sequence:

CSF(k,n,x)

Parameters:

k	-	order of complete symmetric function
n	-	number of variables k <= n , n nonnegint, k integer)
x	-	base name of variables (i.e. we have variables x[1],x[2],..,x[n])

Description:

- The function CSF computes the kth complete symmetric function of the variables x[1],x[2],..,x[n]. The kth complete symmetric function of the variables x[1],x[2],..,x[n]. is defined recursively as: CSF(k,n,x) = x[n]*CSF(k-1,n,x) + CSF(k,n-1,x) with CSF(0,n)=1 and CSF(k,n)=0 for k negint.
- This function is part of the ALTERNANT package and so can be used in the form CSF(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,CSF).

Examples:

```
> with(ALTERNANT):
> CSF(2,3,x);
```

$$x_3^2 + x_3\,x_2 + x_3\,x_1 + x_2^2 + x_2\,x_1 + x_1^2$$

```
> CSF(4,2,x);
```

$$x_2^4 + x_2^3\,x_1 + x_2^2\,x_1^2 + x_2\,x_1^3 + x_1^4$$

```
> CSF(3,3,x);
```

$$x_3^3 + x_3^2\,x_2 + x_3^2\,x_1 + x_3\,x_2^2 + x_1\,x_2\,x_3 + x_3\,x_1^2 + x_2^3 + x_2^2\,x_1 + x_2\,x_1^2 + x_1^3$$

```
> CSF(1,5,x);
```

$$x_5 + x_4 + x_3 + x_2 + x_1$$

```
> CSF(0,7,x);
```

$$1$$

```
> CSF(-1,5,x);
```

$$0$$

See Also: **ALTERNANT[ESF], ALTERNANT[Alternant]**

Function: ALTERNANT[evalcsf] - evaluate an expression containing unevaluated complete symmetric function calls csf(..)

Calling Sequence:

evalcsf(expr)

Parameters:

expr - expression containing unevaluated esf(..) elementary symmetric function calls

Description:

- The function evalcsf computes the expansion of an expression containing unevaluated elementary symmetric function calls. Each occurrence of csf(k,n,x) is replaced with the evaluated function CSF(k,n,x).
- This function is part of the ALTERNANT package and so can be used in the form evalcsf(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,evalcsf).

Examples:

```
> with(ALTERNANT):
> evalcsf(csf(2,4,x));
```

$$x_4{}^2 + x_4\,x_3 + x_4\,x_2 + x_4\,x_1 + x_3{}^2 + x_3\,x_2 + x_3\,x_1 + x_2{}^2 + x_2\,x_1 + x_1{}^2$$

```
> evalcsf(csf(3,4,y));
```

$$y_4{}^3 + y_4{}^2\,y_3 + y_4{}^2\,y_2 + y_4{}^2\,y_1 + y_4\,y_3{}^2 + y_4\,y_3\,y_2 + y_4\,y_3\,y_1 + y_4\,y_2{}^2 + y_4\,y_2\,y_1 + y_4\,y_1{}^2 + y_3{}^3 + y_3{}^2\,y_2 + y_3{}^2\,y_1 + y_3\,y_2{}^2 + y_3\,y_2\,y_1 + y_3\,y_1{}^2 + y_2{}^3 + y_2{}^2\,y_1 + y_2\,y_1{}^2 + y_1{}^3$$

```
> evalcsf(csf(4,3,x));
```

$$x_3{}^4 + x_3{}^3\,x_2 + x_3{}^3\,x_1 + x_3{}^2\,x_2{}^2 + x_3{}^2\,x_2\,x_1 + x_3{}^2\,x_1{}^2 + x_3\,x_2{}^3 + x_3\,x_2{}^2\,x_1 + x_3\,x_2\,x_1{}^2 + x_3\,x_1{}^3 + x_2{}^4 + x_2{}^3\,x_1 + x_2{}^2\,x_1{}^2 + x_2\,x_1{}^3 + x_1{}^4$$

```
> expr:=Alternant(4,x,[[1..2,a*x[i]^(j-1)],[3.. 4,-x[i]^(j+1)]],csf);
```

$$expr := a^2\,(\mathrm{csf}(2,\,4,\,x)^2 - \mathrm{csf}(3,\,4,\,x)\,\mathrm{csf}(1,\,4,\,x))\,\mathrm{ALTERNANT/DP}(4,\,x,\,x_i)$$

```
> evalcsf(expr);
```

$$a^2((x_4{}^2 + x_4\,x_3 + x_4\,x_2 + x_4\,x_1 + x_3{}^2 + x_3\,x_2 + x_3\,x_1 + x_2{}^2 + x_2\,x_1 + x_1{}^2)^2 - (x_4{}^3 + x_4{}^2\,x_3 + x_4{}^2\,x_2 + x_4{}^2\,x_1 + x_4\,x_3{}^2 + x_4\,x_3\,x_2 + x_4\,x_3\,x_1 + x_4\,x_2{}^2 + x_4\,x_2\,x_1 + x_4\,x_1{}^2 + x_3{}^3 + x_3{}^2\,x_2 + x_3{}^2\,x_1 + x_3\,x_2{}^2 + x_3\,x_2\,x_1 + x_3\,x_1{}^2 + x_2{}^3 + x_2{}^2\,x_1 + x_2\,x_1{}^2 + x_1{}^3)(x_4 + x_3 + x_2 + x_1))\mathrm{ALTERNANT/DP}(4,\,x,\,x_i)$$

See Also: ALTERNANT[CSF], ALTERNANT[Alternant]

Function: ALTERNANT[CSFcofactorMatrix] - cofactor of the difference product of the specified simple alternant (in terms of complete symmetric functions) in matrix form

Calling Sequence:

CSFcofactorMatrix(n,x,F)

Parameters:

n	-	alternant order (symbolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
F	-	list of piecewise function specifications (here we require monomials in x[i] !!) specifiying the alternant (see the Specification for details)

Description:

- The function CSFcofactorMatrix returns the cofactor of the difference product in terms of complete symmetric functions of the specified simple alternant (i.e. generating functions have to be monomials) in matrix form. Possible constant factors of the alternant are displayed separately.
- Note, that for symbolic alternant order, the returned matrix should be used for illustrative purposes only, since the dots "o" used to abbreviate the symbolic case are treated as usual matrix entries by Maple in any further computation.Otherwise, the determinant of the returned matrix multiplied with the possible constant factor and the difference product of the variables x[1],...,x[n] equals the determinant of the alternant.

- This function is part of the ALTERNANT package and so can be used in the form CSFcofactorMatrix(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,CSFcofactorMatrix).

Examples:

```
>  with(ALTERNANT):
>  CSFcofactorMatrix(4,x,[[1..2,x[i]^j],[3..4,x[ i]^(j+1)]]);
```

$$\begin{bmatrix} \mathrm{csf}(1,4,x) & \mathrm{csf}(2,4,x) & \mathrm{csf}(4,4,x) & \mathrm{csf}(5,4,x) \\ 1 & \mathrm{csf}(1,4,x) & \mathrm{csf}(3,4,x) & \mathrm{csf}(4,4,x) \\ 0 & 1 & \mathrm{csf}(2,4,x) & \mathrm{csf}(3,4,x) \\ 0 & 0 & \mathrm{csf}(1,4,x) & \mathrm{csf}(2,4,x) \end{bmatrix}$$

```
>  CSFcofactorMatrix(5,x,[[1..2,a*x[i]^j],[3..4, -x[i]^(j+1)],[5..5,x[i]^3]]);
```

$$\mathit{factor\ of\ the\ corresponding\ determinant:}\ ,\ a^2$$

$$\begin{bmatrix} \mathrm{csf}(1,5,x) & \mathrm{csf}(2,5,x) & \mathrm{csf}(4,5,x) & \mathrm{csf}(5,5,x) & \mathrm{csf}(3,5,x) \\ 1 & \mathrm{csf}(1,5,x) & \mathrm{csf}(3,5,x) & \mathrm{csf}(4,5,x) & \mathrm{csf}(2,5,x) \\ 0 & 1 & \mathrm{csf}(2,5,x) & \mathrm{csf}(3,5,x) & \mathrm{csf}(1,5,x) \\ 0 & 0 & \mathrm{csf}(1,5,x) & \mathrm{csf}(2,5,x) & 1 \\ 0 & 0 & 1 & \mathrm{csf}(1,5,x) & 0 \end{bmatrix}$$

```
>  CSFcofactorMatrix(n,x,[[1..n-2,x[i]^(j-1)],[n -1..n,x[i]^(j+1)]]);
```

$$\begin{bmatrix} 1 & \mathrm{csf}(1,n,x) & o & o & \mathrm{csf}(n-4,n,x) & \mathrm{csf}(n-3,n,x) & \mathrm{csf}(n,n,x) & \mathrm{csf}(n+1,n,x) \\ 0 & 1 & o & o & \mathrm{csf}(n-5,n,x) & \mathrm{csf}(n-4,n,x) & \mathrm{csf}(n-1,n,x) & \mathrm{csf}(n,n,x) \\ 0 & 0 & o & o & \mathrm{csf}(n-6,n,x) & \mathrm{csf}(n-5,n,x) & \mathrm{csf}(n-2,n,x) & \mathrm{csf}(n-1,n,x) \\ o & o & o & o & o & o & o & o \\ o & o & o & o & o & o & o & o \\ 0 & 0 & o & o & 0 & 1 & \mathrm{csf}(3,n,x) & \mathrm{csf}(4,n,x) \\ 0 & 0 & o & o & 0 & 0 & \mathrm{csf}(2,n,x) & \mathrm{csf}(3,n,x) \\ 0 & 0 & o & o & 0 & 0 & \mathrm{csf}(1,n,x) & \mathrm{csf}(2,n,x) \end{bmatrix}$$

See Also: **ALTERNANT[Alternant], ALTERNANT[evalcsf]**

Function: ALTERNANT[ESF] - elementary symmetric function

Calling Sequence:

ESF(k,n,x)

Parameters:

k	-	order of elementary symmetric function
n	-	number of variables (0 <= k <= n ,n integer)
x	-	base name of variables (i.e. we have variables x[1],x[2],..,x[n])

Description:

- The function ESF computes the kth elementary symmetric function of the variables x[1],x[2],..,x[n]. The kth elementary symmetric function of the variables x[1],x[2],..,x[n] is defined as sum_{1<= i_1 < i_2 < ... < i_k <=n} x[i_1]x[i_2]...x[i_k] .
- This function is part of the ALTERNANT package and so can be used in the form ESF(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,ESF).

Examples:

```
> with(ALTERNANT):
> ESF(2,4,x);
```

$$x_1\,x_2 + x_1\,x_3 + x_1\,x_4 + x_2\,x_3 + x_2\,x_4 + x_3\,x_4$$

```
> ESF(3,3,x);
```

$$x_1\,x_2\,x_3$$

```
> ESF(0,8,x);
```

$$1$$

```
> ESF(3,6,x);
```

$$x_1\,x_2\,x_3 + x_1\,x_2\,x_4 + x_1\,x_2\,x_5 + x_1\,x_2\,x_6 + x_1\,x_3\,x_4 + x_1\,x_3\,x_5 + x_1\,x_3\,x_6 + x_1\,x_4\,x_5 + x_1\,x_4\,x_6 + x_1\,x_5\,x_6 \\ + x_2\,x_3\,x_4 + x_2\,x_3\,x_5 + x_2\,x_3\,x_6 + x_2\,x_4\,x_5 + x_2\,x_4\,x_6 + x_2\,x_5\,x_6 + x_3\,x_4\,x_5 + x_3\,x_4\,x_6 + x_3\,x_5\,x_6 + x_4\,x_5\,x_6$$

See Also: **ALTERNANT[CSF], ALTERNANT[Alternant], ALTERNANT[evalesf]**

Function: ALTERNANT[ESFcofactorMatrix] - cofactor of the difference product of the specified simple alternant (in terms of elementary symmetric functions) in matrix form

Calling Sequence:

ESFcofactorMatrix(n,x,F)

Parameters:

n	-	alternant order (symbolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
F	-	list of piecewise function specifications (here we require monomials in x[i] !!) specifiying the alternant (see the Specification for details)

Description:

- The function ESFcofactorMatrix returns the cofactor of the difference product in terms of elementary symmetric functions of the specified simple alternant (i.e. generating functions have to be monomials) in matrix form. Possible factors of the alternant are displayed separately.

- Note, that for symbolic alternant order, the returned matrix should be used for illustrative purposes only, since the dots "o" used to abbreviate the symbolic case are treated as usual matrix entries by Maple in any further computation.Otherwise, the determinant of the returned matrix multiplied with the possible constant factor and the difference product of the variables x[1],...,x[n] equals the determinant formula of the alternant.

- This function is part of the ALTERNANT package and so can be used in the form ESFcofactorMatrix(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,ESFcofactorMatrix).

Examples:

```
> with(ALTERNANT):
> ESFcofactorMatrix(5,x,[[1..3,x[i]^(j-1)],[4..  5,x[i]^(j+1)]]);
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \mathrm{S}(5, 5, x) & \mathrm{S}(4, 5, x) & \mathrm{S}(3, 5, x) & \mathrm{S}(2, 5, x) & \mathrm{S}(1, 5, x) & \mathrm{S}(0, 5, x) & 0 \\ 0 & \mathrm{S}(5, 5, x) & \mathrm{S}(4, 5, x) & \mathrm{S}(3, 5, x) & \mathrm{S}(2, 5, x) & \mathrm{S}(1, 5, x) & \mathrm{S}(0, 5, x) \end{bmatrix}$$

```
> ESFcofactorMatrix(4,x,[[1..3,2*x[i]^(j+3)],[4 ..4,-x[i]^(j-3)]]);
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ \mathrm{S}(4, 4, x) & \mathrm{S}(3, 4, x) & \mathrm{S}(2, 4, x) & \mathrm{S}(1, 4, x) & \mathrm{S}(0, 4, x) & 0 & 0 \\ 0 & \mathrm{S}(4, 4, x) & \mathrm{S}(3, 4, x) & \mathrm{S}(2, 4, x) & \mathrm{S}(1, 4, x) & \mathrm{S}(0, 4, x) & 0 \\ 0 & 0 & \mathrm{S}(4, 4, x) & \mathrm{S}(3, 4, x) & \mathrm{S}(2, 4, x) & \mathrm{S}(1, 4, x) & \mathrm{S}(0, 4, x) \end{bmatrix}$$

```
> ESFcofactorMatrix(n,x,[[1..1,x[i]^(j-1)],[2..  2,x[i]^2],[3..n,j*x[i]^j]]);
```

$$\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & n-3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n-1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n \\
\mathrm{S}(n,n,x) & \mathrm{S}(n-1,n,x) & \mathrm{S}(n-2,n,x) & \mathrm{S}(n-3,n,x) & \mathrm{S}(n-4,n,x) & o & o & \mathrm{S}(3,n,x) & \mathrm{S}(2,n,x) & \mathrm{S}(1,n,x) & \mathrm{S}(0,n,x)
\end{bmatrix}$$

See Also: ALTERNANT[Alternant], ALTERNANT[S_to_esf], ALTERNANT[evalesf]

Function: ALTERNANT[S_to_esf] - replace S(k,n,x) with (-1)^k*esf(k,n,x)

Calling Sequence:

S_to_esf(expr)

Parameters:

expr	-	expression (or matrix) containing unevaluated S(..) functions.

Description:

- The function S_to_esf substitutes the longer form (-1)^k*esf(k,n,x) for the abbreviation S(k,n,x). The abbreviation is often used to make the output more readable. If n is integer then the expanded output for the expression containing elementary symmetric function can be obtained using the ALTERNANT package function evalesf.
- This function is part of the ALTERNANT package and so can be used in the form S_to_esf(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,S_to_esf).

Examples:

```
> with(ALTERNANT):
> S_to_esf( sum(S(l,4,x),l=0..4) );
```

$$\mathrm{esf}(0,\,4,\,x) - \mathrm{esf}(1,\,4,\,x) + \mathrm{esf}(2,\,4,\,x) - \mathrm{esf}(3,\,4,\,x) + \mathrm{esf}(4,\,4,\,x)$$

```
> S_to_esf( S(n-2,n,x)*S(3,n,x)-S(0,n,x) );
```

$$-(-1)^n\,\mathrm{esf}(n-2,\,n,\,x)\,\mathrm{esf}(3,\,n,\,x) - \mathrm{esf}(0,\,n,\,x)$$

```
> expr:=Alternant(n,x,[[1..2,2*x[i]^(j-1)],[3.. n,-x[i]^(j+1)]]);
```

$$\textit{formula valid for }\,,\ 2 < n$$

$$\mathit{expr} := -4\,(-1)^{(n+1)}\,(\mathrm{S}(n-2,\,n,\,x)^2 - \mathrm{S}(n-3,\,n,\,x)\,\mathrm{S}(n-1,\,n,\,x))\,\mathrm{ALTERNANT/DP}(n,\,x,\,x_i)$$

```
> S_to_esf(expr);
```

$$4\,(-1)^n\,(\mathrm{esf}(n-2,\,n,\,x)^2-\mathrm{esf}(n-3,\,n,\,x)\,\mathrm{esf}(n-1,\,n,\,x))\,\mathrm{ALTERNANT/DP}(n,\,x,\,x_i)$$

```
> evalesf(S_to_esf(-S(1,3,x)+S(2,3,x)^2));
```

$$x_1+x_2+x_3+(x_3\,x_1+x_3\,x_2+x_2\,x_1)^2$$

See Also: ALTERNANT[ESFcofactorMatrix], ALTERNANT[ESF], ALTERNANT[evalesf], ALTERNANT[Alternant]

Function: ALTERNANT[evalesf] - evaluate an expression containing unevaluated elementary symmetric function calls esf(..)

Calling Sequence:

evalesf(expr)

Parameters:

expr - expression containing unevaluated esf(..) elementary symmetric function calls

Description:

- The function evalesf computes the expansion of an expression containing unevaluated elementary symmetric function calls. Each occurrence of esf(k,n,x) is replaced with the evaluated function ESF(k,n,x).
- This function is part of the ALTERNANT package and so can be used in the form evalesf(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,evalesf).

Examples:

```
> with(ALTERNANT):
> evalesf(esf(2,4,x));
```

$$x_2\,x_1+x_3\,x_1+x_3\,x_2+x_4\,x_1+x_4\,x_2+x_4\,x_3$$

```
> evalesf(esf(4,5,y)+esf(2,5,y));
```

$$y_2\,y_3\,y_4\,y_5+y_1\,y_3\,y_4\,y_5+y_1\,y_2\,y_4\,y_5+y_1\,y_2\,y_3\,y_5+y_1\,y_2\,y_3\,y_4+y_1\,y_5+y_2\,y_5+y_3\,y_5+y_4\,y_5+y_1\,y_2$$
$$+\,y_1\,y_3+y_2\,y_3+y_1\,y_4+y_2\,y_4+y_3\,y_4$$

```
> expr:=Alternant(5,x,[[1..2,x[i]^j],[3..5,x[i] ^(j+1)]],esf,check);
```

$$\mathit{expr}:=\mathrm{S}(5,\,5,\,x)\,\mathrm{S}(3,\,5,\,x)\,\mathit{ALTERNANT}_{\mathit{DP}}(5,\,x,\,x_i)$$

```
> evalesf(S_to_esf(expr));
```

$$x_1\,x_2\,x_3\,x_4\,x_5$$
$$(x_4\,x_3\,x_1+x_1\,x_3\,x_5+x_1\,x_4\,x_5+x_4\,x_3\,x_2+x_2\,x_3\,x_5+x_2\,x_4\,x_5+x_3\,x_4\,x_5+x_4\,x_2\,x_1+x_1\,x_2\,x_5+x_3\,x_2\,x_1)$$
$$(x_2-x_1)\,(x_3-x_1)\,(x_4-x_1)\,(x_5-x_1)\,(x_3-x_2)\,(x_4-x_2)\,(x_5-x_2)\,(x_4-x_3)\,(x_5-x_3)\,(x_5-x_4)$$

See Also: **ALTERNANT[ESF], ALTERNANT[Alternant] , ALTERNANT[S_to_esf]**

Function: ALTERNANT[DoubleAlternantMatrix] - matrix of a specified double alternant

Calling Sequence:

DoubleAlternantMatrix(n,x,y,F)

Parameters:

n	-	alternant order (symbolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
y	-	variable base name (for variables y[1],..,y[n])
F	-	generating function specifiying the double alternant of the forms: For integer orders, F has to be a polynomial in x[i] and y[j], For symbolic order, we require one of the following forms: - $(a\,x_i + b\,y_j)^k$ with k<=n , a,b integer (or integer variables); - Sum($\sum_{k=1}^{d} c_k\, x_i^{p_k}\, y_j^{q_k}$,l=s..n) with 0<=s<=n , d posint, c[k] integer (variables) and p[k]= l | n-l | a , 0<=a<=n q[k]= l | n-l | b , 0<=b<=n. - Sum($\sum_{k=1}^{d} c_k\, x_i^{p_k}\, y_j^{q_k}$,l=s..n-1) with 0<=s<=n-1 , d posint, c[k] integer (variables) and p[k]= l | n-l-1 | a , 0<=a<=n-1 q[k]= l | n-l-1 | b , 0<=b<=n-1.

Description:

- The function DoubleAlternantMatrix returns the matrix of the specified double alternant.
- For symbolic orders, the resulting matrix should be used for illustraive purposes only, since Maple treats the dots 'o' abbreviating the symbolic order as usual matrix entries.
- This function is part of the ALTERNANT package and so can be used in the form DoubleAlternantMatrix(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,DoubleAlternantMatrix).

Examples:

```
> with(ALTERNANT):
> DoubleAlternantMatrix(4,x,y,Sum(2*x[i]^l*y[j] ^l,l=0..4));
```

$$\begin{bmatrix} \sum_{l=0}^{4}(2\,{x_1}^l\,{y_1}^l) & \sum_{l=0}^{4}(2\,{x_1}^l\,{y_2}^l) & \sum_{l=0}^{4}(2\,{x_1}^l\,{y_3}^l) & \sum_{l=0}^{4}(2\,{x_1}^l\,{y_4}^l) \\ \sum_{l=0}^{4}(2\,{x_2}^l\,{y_1}^l) & \sum_{l=0}^{4}(2\,{x_2}^l\,{y_2}^l) & \sum_{l=0}^{4}(2\,{x_2}^l\,{y_3}^l) & \sum_{l=0}^{4}(2\,{x_2}^l\,{y_4}^l) \\ \sum_{l=0}^{4}(2\,{x_3}^l\,{y_1}^l) & \sum_{l=0}^{4}(2\,{x_3}^l\,{y_2}^l) & \sum_{l=0}^{4}(2\,{x_3}^l\,{y_3}^l) & \sum_{l=0}^{4}(2\,{x_3}^l\,{y_4}^l) \\ \sum_{l=0}^{4}(2\,{x_4}^l\,{y_1}^l) & \sum_{l=0}^{4}(2\,{x_4}^l\,{y_2}^l) & \sum_{l=0}^{4}(2\,{x_4}^l\,{y_3}^l) & \sum_{l=0}^{4}(2\,{x_4}^l\,{y_4}^l) \end{bmatrix}$$

```
> DoubleAlternantMatrix(n,x,y,Sum(x[i]^l*y[j]^l ,l=0..n));
```

$$\begin{bmatrix}
\sum_{l=0}^{n} {x_1}^l {y_1}^l & \sum_{l=0}^{n} {x_1}^l {y_2}^l & \sum_{l=0}^{n} {x_1}^l {y_3}^l & o & o & \sum_{l=0}^{n} {x_1}^l {y_{n-2}}^l & \sum_{l=0}^{n} {x_1}^l {y_{n-1}}^l & \sum_{l=0}^{n} {x_1}^l {y_n}^l \\
\sum_{l=0}^{n} {x_2}^l {y_1}^l & \sum_{l=0}^{n} {x_2}^l {y_2}^l & \sum_{l=0}^{n} {x_2}^l {y_3}^l & o & o & \sum_{l=0}^{n} {x_2}^l {y_{n-2}}^l & \sum_{l=0}^{n} {x_2}^l {y_{n-1}}^l & \sum_{l=0}^{n} {x_2}^l {y_n}^l \\
\sum_{l=0}^{n} {x_3}^l {y_1}^l & \sum_{l=0}^{n} {x_3}^l {y_2}^l & \sum_{l=0}^{n} {x_3}^l {y_3}^l & o & o & \sum_{l=0}^{n} {x_3}^l {y_{n-2}}^l & \sum_{l=0}^{n} {x_3}^l {y_{n-1}}^l & \sum_{l=0}^{n} {x_3}^l {y_n}^l \\
o & o & o & o & o & o & o & o \\
o & o & o & o & o & o & o & o \\
\sum_{l=0}^{n} {x_{n-2}}^l {y_1}^l & \sum_{l=0}^{n} {x_{n-2}}^l {y_2}^l & \sum_{l=0}^{n} {x_{n-2}}^l {y_3}^l & o & o & \sum_{l=0}^{n} {x_{n-2}}^l {y_{n-2}}^l & \sum_{l=0}^{n} {x_{n-2}}^l {y_{n-1}}^l & \sum_{l=0}^{n} {x_{n-2}}^l {y_n}^l \\
\sum_{l=0}^{n} {x_{n-1}}^l {y_1}^l & \sum_{l=0}^{n} {x_{n-1}}^l {y_2}^l & \sum_{l=0}^{n} {x_{n-1}}^l {y_3}^l & o & o & \sum_{l=0}^{n} {x_{n-1}}^l {y_{n-2}}^l & \sum_{l=0}^{n} {x_{n-1}}^l {y_{n-1}}^l & \sum_{l=0}^{n} {x_{n-1}}^l {y_n}^l \\
\sum_{l=0}^{n} {x_n}^l {y_1}^l & \sum_{l=0}^{n} {x_n}^l {y_2}^l & \sum_{l=0}^{n} {x_n}^l {y_3}^l & o & o & \sum_{l=0}^{n} {x_n}^l {y_{n-2}}^l & \sum_{l=0}^{n} {x_n}^l {y_{n-1}}^l & \sum_{l=0}^{n} {x_n}^l {y_n}^l
\end{bmatrix}$$

```
> DoubleAlternantMatrix(n,x,y,(x[i]+y[j])^n);
```

$$\begin{bmatrix}
(x_1+y_1)^n & (x_1+y_2)^n & (x_1+y_3)^n & o & o & (x_1+y_{n-2})^n & (x_1+y_{n-1})^n & (x_1+y_n)^n \\
(x_2+y_1)^n & (x_2+y_2)^n & (x_2+y_3)^n & o & o & (x_2+y_{n-2})^n & (x_2+y_{n-1})^n & (x_2+y_n)^n \\
(x_3+y_1)^n & (x_3+y_2)^n & (x_3+y_3)^n & o & o & (x_3+y_{n-2})^n & (x_3+y_{n-1})^n & (x_3+y_n)^n \\
o & o & o & o & o & o & o & o \\
o & o & o & o & o & o & o & o \\
(x_{n-2}+y_1)^n & (x_{n-2}+y_2)^n & (x_{n-2}+y_3)^n & o & o & (x_{n-2}+y_{n-2})^n & (x_{n-2}+y_{n-1})^n & (x_{n-2}+y_n)^n \\
(x_{n-1}+y_1)^n & (x_{n-1}+y_2)^n & (x_{n-1}+y_3)^n & o & o & (x_{n-1}+y_{n-2})^n & (x_{n-1}+y_{n-1})^n & (x_{n-1}+y_n)^n \\
(x_n+y_1)^n & (x_n+y_2)^n & (x_n+y_3)^n & o & o & (x_n+y_{n-2})^n & (x_n+y_{n-1})^n & (x_n+y_n)^n
\end{bmatrix}$$

See Also: ALTERNANT[DoubleAlternant]

Function: ALTERNANT[DoubleAlternant] - determinant of a specified double alternant

Calling Sequence:

DoubleAlternant(n,x,y,F)
DoubleAlternant(n,x,y,F,print)
DoubleAlternant(n,x,y,F,check)
DoubleAlternant(n,x,y,F,print,check)

Parameters:

n	-	alternant order (symbolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
y	-	variable base name (for variables y[1],..,y[n])
F	-	generating function specifiying the double alternant of the forms: For integer orders, F has to be a polynomial in x[i] and y[j], For symbolic order, we require one of the following forms: - $(a\,x_i + b\,y_j)^k$ with k<=n , a,b integer (or integer variables); - Sum($\sum_{k=1}^{d} c_k\, {x_i}^{p_k}\, {y_j}^{q_k}$,l=s..n) with 0<=s<=n , d posint, c[k] integer (variables) and p[k]= l | n-l | a , 0<=a<=n q[k]= l | n-l | b , 0<=b<=n. - Sum($\sum_{k=1}^{d} c_k\, {x_i}^{p_k}\, {y_j}^{q_k}$,l=s..n-1) with 0<=s<=n-1 , d posint, c[k] integer (variables) and p[k]= l | n-l-1 | a , 0<=a<=n-1 q[k]= l | n-l-1 | b , 0<=b<=n-1.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function DoubleAlternant tries to compute the determinant formula of the specified double alternant. If necessary, the valid range of the formula is displayed.
- For symbolic double alternant oders, it is tried to reduce the specified double alternant to frame form and to solve it with functions of the FRAMEFORMS package. In the symbolic case, it is necessary, that we have at most two rows and columns and one diagonal in the double alternant's cofactor matrix, otherwise, we cannot compute the alternant.
- For integer orders, we simply use the normal det function on the cofactor matrix.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a finite order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified alternant matrix before returning the determinant formula.
- This function is part of the ALTERNANT package and so can be used in the form DoubleAlternant(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,DoubleAlternant).

Examples:

```
> with(ALTERNANT):
> DoubleAlternant(n,x,y,(3*x[i]+2*y[j])^(n-1),c heck);
```

$$(-1)^{\mathrm{floor}(1/2\,n)}\,(\prod_{l\_=1}^{n} \mathrm{binomial}(n-1,\, n-l\_)\, 3^{(n-l\_)}\, 2^{(-1+l\_)})\, \mathit{ALTERNANT}_{DP}(n,\, x,\, x_i)$$
$$\mathit{ALTERNANT}_{DP}(n,\, y,\, y_i)$$

```
> DoubleAlternant(n,x,y,(x[i]+y[j])^(n),check);
```

$$n >= 2$$

$$(-1)^{(n+\mathrm{floor}(1/2\,n))} \left(\prod_{l\_=1}^{n+1} \mathrm{binomial}(n,\, n-l\_+1)\right) \left(\sum_{l\_=2}^{n+2} \frac{\mathrm{S}(n+2-l\_,\, n,\, y)\,\mathrm{S}(-2+l\_,\, n,\, x)}{\mathrm{binomial}(n,\, n+2-l\_)}\right)$$
$$\mathit{ALTERNANT}_{DP}(n,\, x,\, x_i)\, \mathit{ALTERNANT}_{DP}(n,\, y,\, y_i)$$

```
> DoubleAlternant(n,x,y,Sum(2*x[i]^l*y[j]^l,l=0 ..n),check);
```

$$2^n \left(\sum_{l\_=2}^{n+2} \mathrm{S}(-2+l\_,\, n,\, y)\,\mathrm{S}(-2+l\_,\, n,\, x)\right) \mathit{ALTERNANT}_{DP}(n,\, x,\, x_i)\, \mathit{ALTERNANT}_{DP}(n,\, y,\, y_i)$$

```
> DoubleAlternant(n,x,y,Sum(x[i]^l*y[j]^l+a*x[i ]^2*y[j]^l,l=0..n),check);
```

$$n >= 6$$

$$-((1+a)\,(-(\sum_{l\_=2}^{n-1} \mathrm{S}(-2+l\_,\, n,\, y)\,\mathrm{S}(-2+l\_,\, n,\, x)) - \mathrm{S}(n,\, n,\, x)\,\mathrm{S}(n,\, n,\, y) - \mathrm{S}(n-1,\, n,\, x)\,\mathrm{S}(n-1,\, n,\, y))$$
$$- \mathrm{S}(n-2,\, n,\, y)\,(\mathrm{S}(n-2,\, n,\, x) - a\,(\sum_{l\_=2}^{n-1} \mathrm{S}(-2+l\_,\, n,\, x)) - \mathrm{S}(n,\, n,\, x)\,a - \mathrm{S}(n-1,\, n,\, x)\,a))$$
$$\mathit{ALTERNANT}_{DP}(n,\, x,\, x_i)\, \mathit{ALTERNANT}_{DP}(n,\, y,\, y_i)$$

```
> DoubleAlternant(n,x,y,Sum(x[i]^l*y[j]^l+x[i]^ l*y[j]^2+a*x[i]^2*y[j]^l,l=0..n-1),check);
```

$$n >= 3$$

$$(2+2\,a-a\,n)\, \mathit{ALTERNANT}_{DP}(n,\, x,\, x_i)\, \mathit{ALTERNANT}_{DP}(n,\, y,\, y_i)$$

See Also: ALTERNANT[DoubleAlternantMatrix] ALTERNANT[DoubleAlternantCofactorMatrix] ALTERNANT[DP]

Function: ALTERNANT[DoubleAlternantCofactorMatrix] - matrix of the cofactor of the difference products of the specified double alternant

Calling Sequence:

DoubleAlternantCofactorMatrix(n,x,y,F)

Parameters:

n	-	alternant order (symbolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
y	-	variable base name (for variables y[1],..,y[n])
F	-	generating function specifiying the double alternant of the forms: For integer orders, F has to be a polynomial in x[i] and y[j], For symbolic order, we require one of the following forms: - $(a\,x_i + b\,y_j)^k$ with k<=n , a,b integer (or integer variables); - Sum($\sum_{k=1}^{d} c_k\,{x_i}^{p_k}\,{y_j}^{q_k}$,l=s..n) with 0<=s<=n , d posint, c[k] integer (variables) and p[k]= l | n-l | a , 0<=a<=n q[k]= l | n-l | b , 0<=b<=n. - Sum($\sum_{k=1}^{d} c_k\,{x_i}^{p_k}\,{y_j}^{q_k}$,l=s..n-1) with 0<=s<=n-1 , d posint, c[k] integer (variables) and p[k]= l | n-l-1 | a , 0<=a<=n-1 q[k]= l | n-l-1 | b , 0<=b<=n-1.

Description:

- The function DoubleAlternantCofactorMatrix returns the matrix of the cofactor of the difference products of the determinant formula of the specified double alternant. A possible sign factor is multiplied to the first row.

- For symbolic orders, the resulting matrix should be used for illustraive purpuses only, since Maple treats the dots 'o' abbreviating the symbolic order as usual matrix entries.

- The possible sign factor is multiplied to the first row of the matrix, such that for integer orders, we indeed get the cofactor of the difference products DP(n,x,x[i])*DP(n,y,y[i]) if we apply Maples det function on the resulting matrix.

- This function is part of the ALTERNANT package and so can be used in the form DoubleAlternantCofactorMatrix(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,DoubleAlternantCofactorMatrix).

Examples:

```
> with(ALTERNANT):
> DoubleAlternantCofactorMatrix(n,x,y,(x[i]+y[j ])^n);
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -\mathrm{binomial}(n,\,n) & -\mathrm{S}(n,\,n,\,y) \\ 0 & 0 & 0 & 0 & \mathrm{binomial}(n,\,n-1) & 0 & \mathrm{S}(n-1,\,n,\,y) \\ 0 & 0 & 0 & o & 0 & 0 & o \\ 0 & 0 & o & 0 & 0 & 0 & o \\ 0 & o & 0 & 0 & 0 & 0 & o \\ \mathrm{binomial}(n,\,0) & 0 & 0 & 0 & 0 & 0 & \mathrm{S}(0,\,n,\,y) \\ \mathrm{S}(n,\,n,\,x) & \mathrm{S}(n-1,\,n,\,x) & o & o & o & \mathrm{S}(0,\,n,\,x) & 0 \end{bmatrix}$$

```
> DoubleAlternantCofactorMatrix(n,x,y,Sum(x[i]^ l*y[j]^l+y[j]^l*x[i],l=0..n));
```

$$\begin{bmatrix}
-1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -\mathrm{S}(n,\,n,\,y) \\
0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{S}(n-1,\,n,\,y) \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & \mathrm{S}(n-2,\,n,\,y) \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & \mathrm{S}(n-3,\,n,\,y) \\
0 & o & 0 & 0 & o & 0 & 0 & 0 & o \\
0 & o & 0 & 0 & 0 & o & 0 & 0 & o \\
0 & o & 0 & 0 & 0 & 0 & o & 0 & o \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & \mathrm{S}(0,\,n,\,y) \\
\mathrm{S}(n,\,n,\,x) & \mathrm{S}(n-1,\,n,\,x) & \mathrm{S}(n-2,\,n,\,x) & \mathrm{S}(n-3,\,n,\,x) & o & o & o & \mathrm{S}(0,\,n,\,x) & 0
\end{bmatrix}$$

```
> DoubleAlternantCofactorMatrix(5,x,y,sum(x[i]^ (5-l)*y[j]^l+2*x[i]^2*y[j]^l,l=0..5));
```

$$\begin{bmatrix}
0 & 0 & -2 & 0 & 0 & -1 & -\mathrm{S}(5,\,5,\,y) \\
0 & 0 & 2 & 0 & 1 & 0 & \mathrm{S}(4,\,5,\,y) \\
0 & 0 & 2 & 1 & 0 & 0 & \mathrm{S}(3,\,5,\,y) \\
0 & 0 & 3 & 0 & 0 & 0 & \mathrm{S}(2,\,5,\,y) \\
0 & 1 & 2 & 0 & 0 & 0 & \mathrm{S}(1,\,5,\,y) \\
1 & 0 & 2 & 0 & 0 & 0 & \mathrm{S}(0,\,5,\,y) \\
\mathrm{S}(5,\,5,\,x) & \mathrm{S}(4,\,5,\,x) & \mathrm{S}(3,\,5,\,x) & \mathrm{S}(2,\,5,\,x) & \mathrm{S}(1,\,5,\,x) & \mathrm{S}(0,\,5,\,x) & 0
\end{bmatrix}$$

```
> DoubleAlternantCofactorMatrix(5,x,y,sum(x[i]^ l*y[j]^l,l=0..6));
```

$$\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{S}(5,\,5,\,y) & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & \mathrm{S}(4,\,5,\,y) & \mathrm{S}(5,\,5,\,y) \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & \mathrm{S}(3,\,5,\,y) & \mathrm{S}(4,\,5,\,y) \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & \mathrm{S}(2,\,5,\,y) & \mathrm{S}(3,\,5,\,y) \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & \mathrm{S}(1,\,5,\,y) & \mathrm{S}(2,\,5,\,y) \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & \mathrm{S}(0,\,5,\,y) & \mathrm{S}(1,\,5,\,y) \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \mathrm{S}(0,\,5,\,y) \\
\mathrm{S}(5,\,5,\,x) & \mathrm{S}(4,\,5,\,x) & \mathrm{S}(3,\,5,\,x) & \mathrm{S}(2,\,5,\,x) & \mathrm{S}(1,\,5,\,x) & \mathrm{S}(0,\,5,\,x) & 0 & 0 & 0 \\
0 & \mathrm{S}(5,\,5,\,x) & \mathrm{S}(4,\,5,\,x) & \mathrm{S}(3,\,5,\,x) & \mathrm{S}(2,\,5,\,x) & \mathrm{S}(1,\,5,\,x) & \mathrm{S}(0,\,5,\,x) & 0 & 0
\end{bmatrix}$$

See Also: ALTERNANT[DP], ALTERNANT[DoubleAlternant]

Function: ALTERNANT[TrigAlternant] - determinant of a simple trigonometric alternant

Calling Sequence:

TrigAlternant(n,x,F)

Parameters:

n	-	alternant order (symbolic or integer value)
x	-	variable base name (for variables x[1],..,x[n])
f	-	list of the form $\sin((j - k1)\, x_i + d1)$ or $\cos((j - k2)\, x_i + d2)$ with d1,d2 integer (or integer variables), k1 nonnegint and k2 posint.
print	-	optional display directive

Description:

- The function TrigAlternant computes the determinant formula of the specified simple trigonometric alternant.
- The directive print (optional) prints the specified alternant matrix before returning the determinant formula.
- This function is part of the ALTERNANT package and so can be used in the form TrigAlternant(..) only after setting the libname appropriately and performing the command with(ALTERNANT) or with(ALTERNANT,TrigAlternant).

Examples:

```
>  with(ALTERNANT):
>  TrigAlternant(n,x,sin(j*x[i]));
```

$$2\, 2^{(1/2\,(n+1)\,(n-2))} \left(\prod_{l\_=1}^{n} \sin(x_{l\_})\right) \left(\prod_{l\_=1}^{n} \left(\prod_{k\_=l\_+1}^{n} (\cos(x_{k\_}) - \cos(x_{l\_}))\right)\right)$$

```
>  TrigAlternant(n+2,y,cos((j-1)*y[i]+k),print);
```

$$\begin{bmatrix} \cos(k) & \cos(y_1 + k) & o & o & \cos(n\, y_1 + k) & \cos((n+1)\, y_1 + k) \\ \cos(k) & \cos(y_2 + k) & o & o & \cos(n\, y_2 + k) & \cos((n+1)\, y_2 + k) \\ o & o & o & o & o & o \\ o & o & o & o & o & o \\ \cos(k) & \cos(y_{n+1} + k) & o & o & \cos(n\, y_{n+1} + k) & \cos((n+1)\, y_{n+1} + k) \\ \cos(k) & \cos(y_{n+2} + k) & o & o & \cos(n\, y_{n+2} + k) & \cos((n+1)\, y_{n+2} + k) \end{bmatrix}$$

$$4\, 2^{(1/2\,(n+3)\,(n-2))} \left(\prod_{l\_=1}^{n+2} \frac{\cos(k)\cos(y_{l\_}) - \sin(k)\sin(y_{l\_})}{\cos(y_{l\_})}\right) \left(\prod_{l\_=1}^{n+2} \left(\prod_{k\_=l\_+1}^{n+2} (\cos(y_{k\_}) - \cos(y_{l\_}))\right)\right)$$

See Also: ALTERNANT[DP]

B.3 The HESSENBERGandCONTINUANT package

This is the tutorial to the HESSENBERGandCONTINUANT package which is also available as on–line documentation.

Help For: Introduction to the HESSENBERGandCONTINUANT package

Calling Sequence:

function(args)
HESSENBERGandCONTINUANT[function](args)

Description:

- To use a HESSENBERGandCONTINUANT function, either define that function alone using the command with(HESSENBERGandCONTINUANT, function), or define all HESSENBERGandCONTINUANT functions using the command with(HESSENBERGandCONTINUANT). Alternatively, invoke the function using the long form HESSENBERGandCONTINUANT[function]. This long form notation is necessary whenever there is a conflict between a package function name and another function used in the same session.
- The functions available are: Continuant, ContinuantMatrix, HessenbergDet, HessenbergMatrix
- This package provides functions to compute a determinant formula for specified Continuants (determinants of tridiagonal matrices) and Hessenberg matrices. An explicit formula can only obtained for special cases, otherwise a recurrence relation for the determinant is returned.
- For more information on a particular function, see HESSENBERGandCONTINUANT[function].

See Also: **ALTERNANT, FRAMEFORMS, SYMMETRIC**

Function: HESSENBERGandCONTINUANT[ContinuantMatrix] - matrix of a specified continuant

Calling Sequence:

ContinuantMatrix(n, mainDiag, upperDiag, lowerDiag)

Parameters:

n	-	continuant order (symbolic or integer value)
mainDiag	-	(possibly piecewise) specification of the main diagonal. e.g. mainDiag = [[1..p1,f1],[p1+1..p2,f2], ... , [n-p_k +1..n,f_k]] with f_l functions in i, and p_l integer. Short cut for [[1..n,f]] is simply f and short cut for [...,[p..p,f],...] is [...,[p,f],...]
upperDiag	-	specification of the first upper diagonal specification like mainDiag with n-1 as end interval bound)
lowerDiag	-	specification of the first lower diagonal specification like mainDiag with n-1 as end interval bound)

Description:

- The function ContinuantMatrix returns the matrix of the specified continuant. If the order is symbolic then dots "o" are used to abbreviate the symbolic form; thus the returned matrix in this case should be used for illustrative purposes only, since Maple treats the "o"'s as usual matrix entries.

- This function is part of the HESSENBERGandCONTINUANT package and so can be used in the form ContinuantMatrix(..) only after setting the libname appropriately and performing the command with(HESSENBERGandCONTINUANT) or with(HESSENBERGandCONTINUANT,ContinuantMatrix).

Examples:

```
> with(HESSENBERGandCONTINUANT):
> ContinuantMatrix(n,x+y,x,y);
```

$$\begin{bmatrix} x+y & x & 0 & 0 & o & o \\ y & o & o & 0 & 0 & o \\ 0 & o & o & o & 0 & 0 \\ o & 0 & o & o & o & 0 \\ o & 0 & 0 & o & o & x \\ 0 & o & o & 0 & y & x+y \end{bmatrix}$$

```
> ContinuantMatrix(5,[[1..2,c[i]],[3..5,d]], -1, [[1..3,y],[4,z[i]]]);
```

$$\begin{bmatrix} c_1 & -1 & 0 & 0 & 0 \\ y & c_2 & -1 & 0 & 0 \\ 0 & y & d & -1 & 0 \\ 0 & 0 & y & d & -1 \\ 0 & 0 & 0 & z_4 & d \end{bmatrix}$$

See Also: HESSENBERGandCONTINUANT[Continuant]

Function: HESSENBERGandCONTINUANT[Continuant] - determinant of the specified continuant

Calling Sequence:

Continuant(n, mainDiag, upperDiag, lowerDiag)
Continuant(n, mainDiag, upperDiag, lowerDiag, print)
Continuant(n, mainDiag, upperDiag, lowerDiag, check)
Continuant(n, mainDiag, upperDiag, lowerDiag, print,check)

Parameters:

n	-	continuant order (symbolic or integer value)
mainDiag	-	(possibly piecewise) specification of the main diagonal. e.g. mainDiag = [[1..p1,f1],[p1+1..p2,f2], ... , [n-p_k +1..n,f_k]] with f_l functions in i, and p_l integer. Short cut for [[1..n,f]] is simply f and short cut for [...,[p..p,f],...] is [...,[p,f],...]
upperDiag	-	specification of the first upper diagonal specification like mainDiag with n-1 as end interval bound)
lowerDiag	-	specification of the first lower diagonal specification like mainDiag with n-1 as end interval bound)
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function Continuant returns the determinant formula of the specified Continuant (tridiagonal matrix). If no explicit formula can be given, a recurrence relation is returned. In some cases, it is possible to solve that recurrence using Maple's rsolve function.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a finite order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified continuant matrix before returning the determinant formula.
- This function is part of the HESSENBERGandDIAGONAL package and so can be used in the form Continuant(..) only after setting the libname appropriately and performing the command with(HESSENBERGandDIAGONAL) or with(HESSENBERGandDIAGONAL,Continuant).

Examples:

```
> with(HESSENBERGandCONTINUANT):
> Continuant(n,x+y,x,y,print,check);
```

$$\mathit{Matrix}:$$

$$\begin{bmatrix} x+y & x & 0 & 0 & o & o \\ y & o & o & 0 & 0 & o \\ 0 & o & o & o & 0 & 0 \\ o & 0 & o & o & o & 0 \\ o & 0 & 0 & o & o & x \\ 0 & o & o & 0 & y & x+y \end{bmatrix}$$

$$\mathit{Determinant}:$$

$$\frac{-y^{(n+1)} + x^{(n+1)}}{-y + x}$$

```
>  Continuant(n,a,b,c);
```

$$a^n \left(\sum_{k\_=0}^{\mathrm{floor}(1/2\,n)} \mathrm{binomial}(n-k\_,\, k\_)\, c^{k\_}\, b^{k\_}\, (-a^2)^{(-k\_)} \right)$$

```
>  Continuant(n,[[1..2,a[i]],[3..n-1,0],[n,k]],x ,y,print,check);
```

$$\mathit{Matrix}:$$

$$\begin{bmatrix} a_1 & x & 0 & 0 & o & o & 0 & 0 & 0 \\ y & a_2 & x & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & y & 0 & x & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & y & o & o & 0 & 0 & 0 & o \\ 0 & 0 & 0 & o & o & o & 0 & 0 & o \\ o & 0 & 0 & 0 & o & o & o & 0 & 0 \\ o & 0 & 0 & 0 & 0 & o & o & x & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & y & 0 & x \\ 0 & 0 & o & o & 0 & 0 & 0 & y & k \end{bmatrix}$$

$$\mathit{Determinant}:$$

$$\mathit{formula\ valid\ for}:,\ 4 \le n$$

$$[[\frac{(a_1\,a_2 - x\,y)\,k\,(-1)^{(1/2\,n-3/2)}\,(x\,y)^{(1/2\,n-1/2)}}{x\,y} + x\,y\,a_1\,(-1)^{(1/2\,n-5/2)}\,(x\,y)^{(1/2\,n-3/2)},\ n-3=2\,m],$$
$$[-(a_1\,a_2 - x\,y)\,(-1)^{(1/2\,n-2)}\,(x\,y)^{(1/2\,n-1)} - a_1\,k\,(-1)^{(1/2\,n-2)}\,(x\,y)^{(1/2\,n-1)},\ \mathit{otherwise}]]$$

```
>  Continuant(n,a[i],b[i],c[i]);
```

$$\mathrm{RECURRENCE}(\{\mathrm{C}(n) = a_n\,\mathrm{C}(n-1) - b_{n-1}\,c_{n-1}\,\mathrm{C}(n-2),\ \mathrm{C}(0)=1,\ \mathrm{C}(-1)=0\})$$

See Also: HESSENBERGandCONTINUANT[ContinuantMatrix]

Function: HESSENBERGandCONTINUANT[HessenbergMatrix] - specified Hessenberg matrix

Calling Sequence:

HessenbergMatrix(n, diagSpecL)

Parameters:

n	-	order of Hessenberg determinant (symbolic or integer value)
diagSpecL	-	specification of the diagonals -1(lower diagonal), 0 (main diagonal), 1, ..., n-1. in the form [[pos1, specL1] , [pos2, specL2] , ... , [pos_k, specL_k]] with -1 <= pos_l <= n-1 For integer n: specL=[[1..p1,f1],[p1+1..p2,f2], ... , [p_k +1..pos, f_k]] short cut for [[1..pos,f]] is simply f and short cut for [...,[p..p,f],...] is [...,[p,f],...] For symbolic n: specL may only be a function in i.

Description:

- The function HessenbergMatrix returns the specified Hessenberg matrix. If the order is symbolic then dots "o" are used to abbreviate the symbolic form; thus the returned matrix in this case should be used for illustrative purposes only, since Maple treats the "o"'s as usual matrix entries.

- This function is part of the HESSENBERGandCONTINUANT package and so can be used in the form HessenbergMatrix(..) only after setting the libname appropriately and performing the command with(HESSENBERGandCONTINUANT) or with(HESSENBERGandCONTINUANT,HessenbergMatrix).

Examples:

```
> with(HESSENBERGandCONTINUANT):
> HessenbergMatrix(4,[[-1,b], [0,[[1..2,1],[3..4,5]]],[2,a[i]] ]);
```

$$\begin{bmatrix} 1 & 0 & a_1 & 0 \\ b & 1 & 0 & a_2 \\ 0 & b & 5 & 0 \\ 0 & 0 & b & 5 \end{bmatrix}$$

```
> HessenbergMatrix(n,[[-1,b],[0,a[i]],[n-1,z]]) ;
```

$$\begin{bmatrix} a_1 & 0 & 0 & o & o & 0 & z \\ b & a_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & b & o & 0 & 0 & 0 & o \\ 0 & 0 & o & o & 0 & 0 & o \\ o & 0 & 0 & o & o & 0 & 0 \\ o & 0 & 0 & 0 & b & a_{n-1} & 0 \\ 0 & o & o & 0 & 0 & b & a_n \end{bmatrix}$$

See Also: HESSENBERGandCONTINUANT[HessenbergDet]

Function: HESSENBERGandCONTINUANTS[HessenbergDet] - determinant formula of specified Hessenberg matrix

Calling Sequence:

HessenbergDet(n, diagSpecL)
HessenbergDet(n, diagSpecL, print)
HessenbergDet(n, diagSpecL, check)
HessenbergDet(n, diagSpecL, print, check)

Parameters:

n	-	order of Hessenberg determinant (symbolic or integer value)
diagSpecL	-	specification of the diagonals -1(lower diagonal), 0 (main diagonal), 1, ..., n-1. in the form [[pos1, specL1] , [pos2, specL2] , ... , [pos_k, specL_k]] with -1 <= pos_l <= n-1 For integer n: specL=[[1..p1,f1],[p1+1..p2,f2], ... , [p_k +1..pos, f_k]] short cut for [[1..pos,f]] is simply f and short cut for [...,[p..p,f],...] is [...,[p,f],...] For symbolic n: specL may only be a function in i.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function HessenbergDet returns the determinant formula of the specified Hessenberg matrix. If no explicit formula can be given, a recurrence relation is returned. In some cases, it is possible to solve that recurrence using Maple's rsolve function.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified continuant matrix before returning the determinant formula.
- This function is part of the HESSENBERGandCONTINUANT package and so can be used in the form HessenbergDet(..) only after setting the libname appropriately and performing the command with(HESSENBERGandCONTINUANT) or with(HESSENBERGandCONTINUANT,HessenbergDet).

Examples:

```
>  with(HESSENBERGandCONTINUANT):
>  HessenbergDet(n,[[-1,b],[0,a],[n-3,x],[n-2,y] ,[n-1,z]],print,check);
```

$$Matrix:$$

$$\begin{bmatrix} a & 0 & 0 & o & o & 0 & x & y & z \\ b & a & 0 & 0 & 0 & 0 & 0 & x & y \\ 0 & b & a & 0 & 0 & 0 & 0 & 0 & x \\ 0 & 0 & b & o & 0 & 0 & 0 & 0 & 0 \\ o & 0 & 0 & o & o & 0 & 0 & 0 & o \\ o & 0 & 0 & 0 & o & o & 0 & 0 & o \\ 0 & 0 & 0 & 0 & 0 & b & a & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & b & a & 0 \\ 0 & 0 & 0 & o & o & 0 & 0 & b & a \end{bmatrix}$$

$$Determinant:$$

$$\textit{formula valid for}:,\ 5 \le n$$

$$\frac{a^{(n+1)}}{a} + 3\,\frac{(-1)^{(n-3)}\,x\,a^2\,b^{(n-2)}}{b} + 2\,\frac{(-1)^{(n-2)}\,y\,a\,b^{(n-1)}}{b} + \frac{(-1)^{(n-1)}\,z\,b^n}{b}$$

```
>  HessenbergDet(n,[[-1,b],[2,a],[n-3,x],[n-2,y] ,[n-1,z]],print,check);
```

$$\textit{Matrix}:$$

$$\begin{bmatrix} 0 & 0 & a & 0 & 0 & o & o & 0 & x & y & z \\ b & 0 & 0 & a & 0 & 0 & 0 & 0 & 0 & x & y \\ 0 & b & 0 & 0 & a & 0 & 0 & 0 & 0 & 0 & x \\ 0 & 0 & b & 0 & 0 & o & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & o & 0 & 0 & o & 0 & 0 & 0 & o \\ 0 & 0 & 0 & 0 & o & 0 & 0 & o & 0 & 0 & o \\ o & 0 & 0 & 0 & 0 & o & 0 & 0 & a & 0 & 0 \\ o & 0 & 0 & 0 & 0 & 0 & o & 0 & 0 & a & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b & 0 & 0 & a \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b & 0 & 0 \\ 0 & 0 & 0 & o & o & 0 & 0 & 0 & 0 & b & 0 \end{bmatrix}$$

$$\textit{Determinant}:$$

$$\textit{formula valid for}:,\ 6 \le n$$

$$[[(a\,b^2)^{(1/3\,n)} - \frac{z\,b^n\,(-1)^n}{b},\ n = 3\,k],\ [-\frac{z\,b^n\,(-1)^n}{b},\ \textit{otherwise}]]$$

```
>  HessenbergDet(n,[[-1,b],[0,a],[1,c],[2,d]],pr int,check);
```

$$\textit{Matrix}:$$

$$\begin{bmatrix} a & c & d & 0 & 0 & o & o & 0 \\ b & a & c & d & 0 & 0 & 0 & o \\ 0 & b & o & o & o & 0 & 0 & o \\ 0 & 0 & o & o & o & o & 0 & 0 \\ 0 & 0 & 0 & o & o & o & o & 0 \\ o & 0 & 0 & 0 & o & o & o & d \\ o & 0 & 0 & 0 & 0 & o & o & c \\ 0 & o & o & 0 & 0 & 0 & b & a \end{bmatrix}$$

$$\textit{Determinant}:$$

$$\textit{formula valid for}:,\ 3 \le n$$

RECURRENCE(

$\{\mathrm{HB}(-2..-1) = 0,\ \mathrm{HB}(n) = a\,\mathrm{HB}(n-1) - c\,b\,\mathrm{HB}(n-2) + d\,b^2\,\mathrm{HB}(n-3),\ \mathrm{HB}(0) = 1\})$

```
> HessenbergDet(5,[[-1,b],[0,[[1..2,a[i]],[3..5 ,i]]],[1,[[1,c],[2..4,-1]]],[2,d],[4,e]],
print,check);
```

$$\textit{Matrix}:$$

$$\begin{bmatrix} a_1 & c & d & 0 & e \\ b & a_2 & -1 & d & 0 \\ 0 & b & 3 & -1 & d \\ 0 & 0 & b & 4 & -1 \\ 0 & 0 & 0 & b & 5 \end{bmatrix}$$

$$\textit{Determinant}:$$

$$60\,a_1\,a_2 + 8\,a_1\,a_2\,b + a_1\,a_2\,d\,b^2 + 20\,a_1\,b + a_1\,b^2 + 5\,a_1\,d\,b^2 - 60\,c\,b - 8\,c\,b^2 - c\,d\,b^3 + 20\,d\,b^2 + d\,b^3 + e\,b^4$$

See Also: HESSENBERGandCONTINUANT[Continuant], HESSENBERGandCONTINUANT[HessenbergMatrix]

B.4 The SYMMETRIC package

This is the tutorial to the SYMMETRIC package which is also available as on–line documentation.

Help For: Introduction to the SYMMETRIC package

Calling Sequence:

function(args)
SYMMETRIC[function](args)

Description:

- To use a SYMMETRIC function, either define that function alone using the command with(SYMMETRIC, function), or define all SYMMETRIC functions using the command with(SYMMETRIC). Alternatively, invoke the function using the long form SYMMETRIC[function]. This long form notation is necessary whenever there is a conflict between a package function name and another function used in the same session.

- The functions available are: AxisymmetricDet, AxisymmetricMatrix, CentrosymmetricDet, CentrosymmetricMatrix, Circulant, CirculantMatrix, ToeplitzDet, ToeplitzMatrix

- This package provides a number of functions specifying and computing the determinant of matrices that obey a certain symmetry like axisymmetry (with respect to the main diagonal), centrosymmetry (with respect to the central element of the matrix) and persymmetry (with respect to the counter main diagonal). It is possible to derive determinant formulas for arbitrary orders for special cases of symmetric matrices with polynomial entries using these functions.

- For more information on a particular function, see ?SYMMETRIC[function].

See Also: **ALTERNANT , FRAMEFORMS , HESSENBERGandCONTINUANT**

Function: SYMMETRIC[AxisymmetricMatrix] - specified axisymmetric matrix

Calling Sequence:

AxisymmetricMatrix(n, mainDiag, restDiags)
AxisymmetricMatrix(n, mainDiag, restDiags, skew)

Parameters:

n	-	matrix order (symbolic or integer value)
mainDiag	-	total function in i specifying the main diagonal, i.e. $a_{i,i}$ for 1 <= i <= n
restDiags	-	total function in i and j specifying the matrix elements $a_{i,j} = a_{j,i}$ for 1 <= i < j <= n
skew	-	(optional) directive for skewsymmetry, i.e. $a_{i,j} = -a_{j,i}$.

Description:

- The function AxisymmetricMatrix returns the specified axisymmetric matrix. If the order is symbolic then dots "o" are used to abbreviate the symbolic form; thus the returned matrix in this case should be used for illustrative purposes only, since Maple treats the "o"'s as usual matrix entries.
- This function is part of the SYMMETRIC package and so can be used in the form AxisymmetricMatrix(..) only after setting the libname appropriately and performing the command with(SYMMETRIC) or with(SYMMETRIC,AxisymmetricMatrix).

Examples:

```
>  with(SYMMETRIC):
>  AxisymmetricMatrix(n,a[i],b);
```

$$\begin{bmatrix} a_1 & b & b & o & o & b & b & b \\ b & a_2 & b & b & o & o & b & b \\ b & b & a_3 & o & o & o & o & b \\ o & b & o & o & o & o & o & o \\ o & o & o & o & o & o & b & o \\ b & o & o & o & o & a_{n-2} & b & b \\ b & b & o & o & o & b & a_{n-1} & b \\ b & b & b & o & o & b & b & a_n \end{bmatrix}$$

```
>  AxisymmetricMatrix(5,i+i-1,i+j-1);
```

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 5 & 6 & 7 \\ 4 & 5 & 6 & 7 & 8 \\ 5 & 6 & 7 & 8 & 9 \end{bmatrix}$$

```
>  AxisymmetricMatrix(4,x[i],x[i]*x[j],skew);
```

$$\begin{bmatrix} x_1 & x_1\,x_2 & x_1\,x_3 & x_1\,x_4 \\ -x_1\,x_2 & x_2 & x_2\,x_3 & x_2\,x_4 \\ -x_1\,x_3 & -x_2\,x_3 & x_3 & x_3\,x_4 \\ -x_1\,x_4 & -x_2\,x_4 & -x_3\,x_4 & x_4 \end{bmatrix}$$

See Also: AxisymmetricDet

Function: SYMMETRIC[AxisymmetricDet] -determinant of specified axisymmetric matrix

Calling Sequence:

AxisymmetricDet(n, mainDiag, restDiags)
AxisymmetricDet(n, mainDiag, restDiags, skew)

AxisymmetricDet(n, mainDiag, restDiags, print)
AxisymmetricDet(n, mainDiag, restDiags, check)
AxisymmetricDet(n, mainDiag, restDiags, skew, print)
AxisymmetricDet(n, mainDiag, restDiags, skew, check)
AxisymmetricDet(n, mainDiag, restDiags, print, check)
AxisymmetricDet(n, mainDiag, restDiags, skew, print, check)

Parameters:

n	-	matrix order (symbolic or integer value)
mainDiag	-	total function in i specifying the main diagonal, i.e. $a_{i,\,i}$ for 1 <= i <= n
restDiags	-	total function in i and j specifying the matrix elements $a_{i,\,j} = a_{j,\,i}$ for 1 <= i < j <= n
skew	-	(optional) directive for skewsymmetry, i.e. $a_{i,\,j} = -a_{j,\,i}$.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function AxisymmetricDet returns a determinant formula for the specified axisymmetric matrix. If the specification is too general then the unevaluated function call is returned for symbolic orders.In some cases, only a recurrence relation can be returned.

- If the optional directive skew is given then the corresponding skewsymmetric determinant is computed.

- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order `n` is an integer, the check value is set to `n`.

- The directive print (optional) prints the specified alternant matrix before returning the determinant formula.

- This function is part of the SYMMETRIC package and so can be used in the form AxisymmetricDet(..) only after setting the libname appropriately and performing the command with(SYMMETRIC) or with(SYMMETRIC,AxisymmetricDet).

Examples:

```
> with(SYMMETRIC):

> AxisymmetricDet(n,a[i],b,print,check);
```

Matrix :

$$\begin{bmatrix} a_1 & b & b & o & o & b & b & b \\ b & a_2 & b & b & o & o & b & b \\ b & b & a_3 & o & o & o & o & b \\ o & b & o & o & o & o & o & o \\ o & o & o & o & o & o & b & o \\ b & o & o & o & o & a_{n-2} & b & b \\ b & b & o & o & b & b & a_{n-1} & b \\ b & b & b & o & o & b & b & a_n \end{bmatrix}$$

$$Determinant:$$

$$\left(\prod_{l\_=1}^{n} (a_{l\_} - b)\right) \left(1 + \left(\sum_{k\_=1}^{n} \frac{b}{a_{k\_} - b}\right)\right)$$

```
>   AxisymmetricDet(n,x[i],x[i]*x[j],check);
```

$$\left(\prod_{l\_=1}^{n} (x_{l\_} - {x_{l\_}}^2)\right) \left(1 + \left(\sum_{l\_=1}^{n} \frac{{x_{l\_}}^2}{x_{l\_} - {x_{l\_}}^2}\right)\right)$$

```
>   AxisymmetricDet(n,a[i],b[i],print,check);
```

$$Matrix:$$

$$\begin{bmatrix} a_1 & b_1 & b_1 & o & o & b_1 & b_1 & b_1 \\ b_1 & a_2 & b_2 & b_2 & o & o & b_2 & b_2 \\ b_1 & b_2 & a_3 & o & o & o & o & b_3 \\ o & b_2 & o & o & o & o & o & o \\ o & o & o & o & o & o & b_{n-3} & o \\ b_1 & o & o & o & o & a_{n-2} & b_{n-2} & b_{n-2} \\ b_1 & b_2 & o & o & b_{n-3} & b_{n-2} & a_{n-1} & b_{n-1} \\ b_1 & b_2 & b_3 & o & o & b_{n-2} & b_{n-1} & a_n \end{bmatrix}$$

$$Determinant:$$

RECURRENCE(

$$\{\mathrm{C}(1) = a_1,\ \mathrm{C}(0) = 1,\ \mathrm{C}(n) = (a_{n-1} + 2\,b_{n-1} + a_n)\,\mathrm{C}(n-1) + (a_{n-1} - b_{n-1})^2\,\mathrm{C}(n-2)\})$$

```
>   AxisymmetricDet(n,k,2*(i-j)+k,print,check);
```

$$Matrix:$$

$$\begin{bmatrix} k & -2+k & -4+k & o & o & 6-2n+k & 4-2n+k & 2-2n+k \\ -2+k & k & -2+k & -4+k & o & o & 6-2n+k & 4-2n+k \\ -4+k & -2+k & k & o & o & o & o & 6-2n+k \\ o & -4+k & o & o & o & o & o & o \\ o & o & o & o & o & o & -2+k & o \\ 6-2n+k & o & o & o & o & k & -2+k & -2+k \\ 4-2n+k & 6-2n+k & o & o & -2+k & -2+k & k & -2+k \\ 2-2n+k & 4-2n+k & 6-2n+k & o & o & -2+k & -2+k & k \end{bmatrix}$$

Determinant :

$$-2\,(-1)^{(n-1)}\,(-4)^{(n-2)}\,(-2\,n+2+2\,k)$$

```
>  AxisymmetricDet(n,a[i],b[i],skew,print,check) ;
```

Matrix :

$$\begin{bmatrix} a_1 & b_1 & b_1 & o & o & b_1 & b_1 & b_1 \\ -b_1 & a_2 & b_2 & b_2 & o & o & b_2 & b_2 \\ -b_1 & -b_2 & a_3 & o & o & o & o & b_3 \\ o & -b_2 & o & o & o & o & o & o \\ o & o & o & o & o & o & b_{n-3} & o \\ -b_1 & o & o & o & o & a_{n-2} & b_{n-2} & b_{n-2} \\ -b_1 & -b_2 & o & o & -b_{n-3} & -b_{n-2} & a_{n-1} & b_{n-1} \\ -b_1 & -b_2 & -b_3 & o & o & -b_{n-2} & -b_{n-1} & a_n \end{bmatrix}$$

Determinant :

RECURRENCE(
$\{\mathrm{C}(1)=a_1,\ \mathrm{C}(0)=1,\ \mathrm{C}(n)=(a_n+a_{n-1})\,\mathrm{C}(n-1)+({a_{n-1}}^2-{b_{n-1}}^2)\,\mathrm{C}(n-2)\})$

```
>  AxisymmetricDet(n,0,x[i,j],skew,check);
```

$$[[\text{SYMMETRIC/Pfaffian}(n,\ x_{i,\,j})^2,\ n=2\,m],\ [0,\ \textit{otherwise}]]$$

```
>  AxisymmetricDet(4,1,x[i,j],skew,check);
```

$$(x_{1,\,2}\,x_{3,\,4}-x_{1,\,3}\,x_{2,\,4}+x_{1,\,4}\,x_{2,\,3})^2+{x_{3,\,4}}^2+{x_{2,\,4}}^2+{x_{2,\,3}}^2+{x_{1,\,4}}^2+{x_{1,\,3}}^2+{x_{1,\,2}}^2+1$$

See Also: AxisymmetricMatrix

Function: SYMMETRIC[CentrosymmetricMatrix] - specified centrosymmetric matrix

Calling Sequence:

CentrosymmetricDet(n, f)

Parameters:

n	-	matrix order (positive integer)
f	-	function in i specifying the matrix elements up to the central element (read rowwise).

Description:

- The function CentrosymmetricMatrix returns the specified centrosymmetric matrix (a matrix that is symmetric with respect to its central element: the r th row reversed is the n-r+1 th row, i.e. the the matrix is the same when read backwards as when read forwards).
- This function is part of the SYMMETRIC package and so can be used in the form CentrosymmetricMatrix(..) only after setting the libname appropriately and performing the command with(SYMMETRIC) or with(SYMMETRIC,CentrosymmetricMatrix).

Examples:

```
> with(SYMMETRIC):
> CentrosymmetricMatrix(4,a[i]);
```

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ a_5 & a_6 & a_7 & a_8 \\ a_8 & a_7 & a_6 & a_5 \\ a_4 & a_3 & a_2 & a_1 \end{bmatrix}$$

```
> CentrosymmetricMatrix(5,i+x);
```

$$\begin{bmatrix} 1+x & 2+x & 3+x & 4+x & 5+x \\ 6+x & 7+x & 8+x & 9+x & 10+x \\ 11+x & 12+x & 13+x & 12+x & 11+x \\ 10+x & 9+x & 8+x & 7+x & 6+x \\ 5+x & 4+x & 3+x & 2+x & 1+x \end{bmatrix}$$

See Also: **CentrosymmetricDet**

Function: SYMMETRIC[CentrosymmetricDet] - determinant of specified centrosymmetric matrix

Calling Sequence:

CentrosymmetricDet(n, f)
CentrosymmetricDet(n, f, print)
CentrosymmetricDet(n, f, check)
CentrosymmetricDet(n, f, print, check)

Parameters:

n	-	matrix order (positive integer).
f	-	function in i specifying the matrix elements up to the central element (read rowwise).
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer).

Description:

- The function CentrosymmetricDet returns the determinant of the specified centrosymmetric matrix (a matrix that is symmetric with respect to its central element: the r th row reversed is the n-r+1 th row, i.e. the the matrix is the same when read backwards as when read forwards). The computation makes use of the special structure and is faster than applying the normal det function.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of the specified matrix is computed using the standard det function and is compared with the value of the output formula.
- The directive print (optional) prints the specified alternant matrix before returning the determinant formula.
- This function is part of the SYMMETRIC package and so can be used in the form CentrosymmetricDet(..) only after setting the libname appropriately and performing the command with(SYMMETRIC) or with(SYMMETRIC,CentrosymmetricDet).

Examples:

```
> with(SYMMETRIC):
> CentrosymmetricDet(4,a[i],print,check);
```

$$\mathit{Matrix}:$$

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ a_5 & a_6 & a_7 & a_8 \\ a_8 & a_7 & a_6 & a_5 \\ a_4 & a_3 & a_2 & a_1 \end{bmatrix}$$

$$\mathit{Determinant}:$$

$$(a_5\,a_2 - a_7\,a_4 + a_5\,a_3 - a_7\,a_1 + a_8\,a_2 - a_6\,a_4 + a_8\,a_3 - a_6\,a_1)$$
$$(a_5\,a_2 - a_7\,a_4 - a_5\,a_3 + a_7\,a_1 - a_8\,a_2 + a_6\,a_4 + a_8\,a_3 - a_6\,a_1)$$

```
> CentrosymmetricDet(5,i+x[i],print,check);
```

$$\mathit{Matrix}:$$

$$\begin{bmatrix} 1+x_1 & 2+x_2 & 3+x_3 & 4+x_4 & 5+x_5 \\ 6+x_6 & 7+x_7 & 8+x_8 & 9+x_9 & 10+x_{10} \\ 11+x_{11} & 12+x_{12} & 13+x_{13} & 12+x_{12} & 11+x_{11} \\ 10+x_{10} & 9+x_9 & 8+x_8 & 7+x_7 & 6+x_6 \\ 5+x_5 & 4+x_4 & 3+x_3 & 2+x_2 & 1+x_1 \end{bmatrix}$$

$$\mathit{Determinant}:$$

$$\begin{aligned}&-((6+x_6)\,(2+x_2)-(5+x_5)\,(9+x_9)-(2+x_2)\,(10+x_{10})+(5+x_5)\,(7+x_7)\\&\quad-(4+x_4)\,(6+x_6)+(1+x_1)\,(9+x_9)+(4+x_4)\,(10+x_{10})-(7+x_7)\,(1+x_1))(-2\,x_8\,x_5\,x_{12}\\&\quad-2\,x_3\,x_{11}\,x_7+2\,x_4\,x_8\,x_{11}+2\,x_8\,x_{11}\,x_2+x_{13}\,x_7\,x_1+2\,x_3\,x_{12}\,x_6-2\,x_3\,x_9\,x_{11}-2\,x_8\,x_{12}\,x_1\\&\quad-x_4\,x_{13}\,x_6-32\,x_4-32\,x_2+16\,x_1+32\,x_3+16\,x_5-6\,x_6+12\,x_7-12\,x_8+12\,x_9-6\,x_{10}\\&\quad-x_{13}\,x_6\,x_2+x_{13}\,x_9\,x_1-x_{13}\,x_{10}\,x_2+2\,x_3\,x_{10}\,x_{12}+13\,x_7\,x_1-13\,x_6\,x_2-16\,x_{12}\,x_1+16\,x_{11}\,x_2\\&\quad+6\,x_{12}\,x_6-6\,x_{11}\,x_7+16\,x_{13}\,x_1+6\,x_{13}\,x_7-16\,x_{13}\,x_2-6\,x_{13}\,x_6-24\,x_8\,x_1-12\,x_8\,x_{12}+22\,x_8\,x_2\\&\quad+12\,x_8\,x_{11}+24\,x_3\,x_6+32\,x_3\,x_{12}-22\,x_3\,x_7-32\,x_3\,x_{11}+13\,x_9\,x_1-13\,x_{10}\,x_2-22\,x_3\,x_9\\&\quad+24\,x_3\,x_{10}-6\,x_9\,x_{11}+6\,x_{10}\,x_{12}+6\,x_{13}\,x_9-6\,x_{13}\,x_{10}-13\,x_4\,x_6+22\,x_4\,x_8-13\,x_4\,x_{10}\\&\quad+16\,x_4\,x_{11}-16\,x_4\,x_{13}-x_4\,x_{13}\,x_{10}+x_{13}\,x_5\,x_7+x_9\,x_{13}\,x_5+13\,x_5\,x_7-24\,x_8\,x_5-16\,x_5\,x_{12}\\&\quad+16\,x_{13}\,x_5+13\,x_9\,x_5)\end{aligned}$$

See Also: **CentrosymmetricMatrix**

Function: SYMMETRIC[CirculantMatrix] - specified Circulant matrix

Calling Sequence:

CirculantMatrix(n, specL)
CirculantMatrix(n, specL, leftshift)

Parameters:

n	-	matrix order (symbolic or integer value)
specL	-	list specifying the first row of the circulant of the form: [[interval1, f1], [interval2, f2] , ...] where intervals are of the form p1..p2 (with p1<=p2) and must be a partition of [1..n] [p1..p2, f] means that element i (p1<= i <= p2) is generated by f(i). Short Cuts: f instead of [[1..n, f]] and [pos, f] instead of [pos..pos, f] . The other rows of the circulant matrix are obtained by "right shifting".
leftshift	-	(optional) directive to use "left shifting" instead of right shifting to get the other rows.

Description:

- The function CirculantMatrix returns the matrix of the specified Circulant (a circulant matrix is a matrix such that any row is got from the preceeding row by passing the last element over the others to the first place). If the order is symbolic then dots "o" are used to abbreviate the symbolic form; thus the returned matrix in this case should be used for illustrative purposes only, since Maple treats the "o"'s as usual matrix entries.
- This function is part of the SYMMETRIC package and so can be used in the form CirculantMatrix(..) only after setting the libname appropriately and performing the command with(SYMMETRIC) or with(SYMMETRIC,CirculantMatrix).

Examples:

```
>  with(SYMMETRIC):
>  CirculantMatrix(n,a[i]);
```

$$\begin{bmatrix} a_1 & a_2 & o & o & a_n \\ a_n & a_1 & a_2 & o & o \\ o & a_n & a_1 & a_2 & o \\ o & o & a_n & a_1 & a_2 \\ a_2 & o & o & a_n & a_1 \end{bmatrix}$$

```
>  CirculantMatrix(n,a[i],leftshift);
```

$$\begin{bmatrix} a_1 & a_2 & o & o & a_n \\ a_2 & o & o & a_n & a_1 \\ o & o & a_n & a_1 & a_2 \\ o & a_n & a_1 & a_2 & o \\ a_n & a_1 & a_2 & o & o \end{bmatrix}$$

```
>  CirculantMatrix(n,[[1..3,a],[4..n,b]]);
```

$$\begin{bmatrix} a & a & a & b & b & o & o & b \\ b & a & a & a & b & b & o & o \\ b & b & a & a & a & b & b & o \\ o & b & b & a & a & a & b & b \\ o & o & b & b & a & a & a & b \\ b & o & o & b & b & a & a & a \\ a & b & o & o & b & b & a & a \\ a & a & b & o & o & b & b & a \end{bmatrix}$$

```
>  CirculantMatrix(5,[[1..2,a+i],[3,x^2],[4..5,p [i]]]);
```

$$\begin{bmatrix} a+1 & a+2 & x^2 & p_4 & p_5 \\ p_5 & a+1 & a+2 & x^2 & p_4 \\ p_4 & p_5 & a+1 & a+2 & x^2 \\ x^2 & p_4 & p_5 & a+1 & a+2 \\ a+2 & x^2 & p_4 & p_5 & a+1 \end{bmatrix}$$

See Also: Circulant

Function: SYMMETRIC[Circulant] - determinant of specified circulant

Calling Sequence:

CirculantMatrix(n, specL)
CirculantMatrix(n, specL, leftshift)
CirculantMatrix(n, specL, print)
CirculantMatrix(n, specL, check)
CirculantMatrix(n, specL, leftshift, print)

CirculantMatrix(n, specL, leftshift, check)
CirculantMatrix(n, specL, print, check)
CirculantMatrix(n, specL, leftshift, print, check)

Parameters:

n	-	matrix order (symbolic or integer value)
specL	-	list specifying the first row of the circulant of the form: [[interval1, f1], [interval2, f2] , ...] where intervals are of the form p1..p2 (with p1<=p2) and must be a partition of [1..n] [p1..p2, f] means that element i (p1<= i <= p2) is generated by f(i). Short Cuts: f instead of [[1..n, f]] and [pos, f] instead of [pos..pos, f] . The other rows of the circulant matrix are obtained by "right shifting".
leftshift	-	(optional) directive to use "left shifting" instead of right shifting to get the other rows.
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function Circulant returns a determinant formula for the specified circulant (a circulant matrix is a matrix such that any row is got from the preceeding row by passing the last element over the others to the first place). For some special cases of circulants, it is possible to derive nice formulas whereas for most cases we have to settle with a general formula involving roots of unity.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified alternant matrix before returning the determinant formula.
- This function is part of the SYMMETRIC package and so can be used in the form Circulant(..) only after setting the libname appropriately and performing the command with(SYMMETRIC) or with(SYMMETRIC,Circulant).

Examples:

```
> with(SYMMETRIC):
> Circulant(n,a[i],print,check);
```

$$Matrix:$$

$$\begin{bmatrix} a_1 & a_2 & o & o & a_n \\ a_n & a_1 & a_2 & o & o \\ o & a_n & a_1 & a_2 & o \\ o & o & a_n & a_1 & a_2 \\ a_2 & o & o & a_n & a_1 \end{bmatrix}$$

$$Determinant:$$

$$\prod_{k\_=1}^{n}\left(\sum_{l\_=1}^{n}(\cos(2\,\frac{\pi\,(l\_-1)\,k\_}{n})+I\sin(2\,\frac{\pi\,(l\_-1)\,k\_}{n}))\,a_{l\_}\right)$$

```
>  Circulant(n,a[i],leftshift,print,check);
```

$$Matrix:$$

$$\begin{bmatrix} a_1 & a_2 & o & o & a_n \\ a_2 & o & o & a_n & a_1 \\ o & o & a_n & a_1 & a_2 \\ o & a_n & a_1 & a_2 & o \\ a_n & a_1 & a_2 & o & o \end{bmatrix}$$

$$Determinant:$$

$$(-1)^{(1/2\,(n-1)\,(n-2))}\left(\prod_{k\_=1}^{n}\left(\sum_{l\_=1}^{n}(\cos(2\,\frac{\pi\,(l\_-1)\,k\_}{n})+I\sin(2\,\frac{\pi\,(l\_-1)\,k\_}{n}))\,a_{l\_}\right)\right)$$

```
>  Circulant(n,[[1..3,a],[4..n,b]],print,check);
```

$$Matrix:$$

$$\begin{bmatrix} a & a & a & b & b & o & o & b \\ b & a & a & a & b & b & o & o \\ b & b & a & a & a & b & b & o \\ o & b & b & a & a & a & b & b \\ o & o & b & b & a & a & a & b \\ b & o & o & b & b & a & a & a \\ a & b & o & o & b & b & a & a \\ a & a & b & o & o & b & b & a \end{bmatrix}$$

$$Determinant:$$

$$formula\ valid\ for:,\ 4\leq n$$

$$[[(3\,a+(n-3)\,b)\,(a-b)^{(n-1)},\ \mathrm{GCD}(3,\,n-3)=1],\ [0,\ otherwise]]$$

```
>  Circulant(n,k*x^(2*i+1),print,check);
```

$$Matrix:$$

$$\begin{bmatrix} k\,x^3 & k\,x^5 & o & o & k\,x^{(2\,n+1)} \\ k\,x^{(2\,n+1)} & k\,x^3 & k\,x^5 & o & o \\ o & k\,x^{(2\,n+1)} & k\,x^3 & k\,x^5 & o \\ o & o & k\,x^{(2\,n+1)} & k\,x^3 & k\,x^5 \\ k\,x^5 & o & o & k\,x^{(2\,n+1)} & k\,x^3 \end{bmatrix}$$

$$Determinant:$$

$$(k\,x^3)^n\,(1-(x^2)^n)^{(n-1)}$$

```
>  Circulant(4,[[1..2,a[i]],[3..4,b[i]]],print,c heck);
```

$$\mathit{Matrix}\ :$$

$$\begin{bmatrix} a_1 & a_2 & b_3 & b_4 \\ b_4 & a_1 & a_2 & b_3 \\ b_3 & b_4 & a_1 & a_2 \\ a_2 & b_3 & b_4 & a_1 \end{bmatrix}$$

$$\mathit{Determinant}\ :$$

$$(a_1+a_2+b_3+b_4)\,(a_1-a_2+b_3-b_4)\,({a_1}^2-2\,b_3\,a_1+{a_2}^2-2\,a_2\,b_4+{b_3}^2+{b_4}^2)$$

See Also: CirculantMatrix

Function: SYMMETRIC[ToeplitzMatrix] -specified Toeplitz matrix

Calling Sequence: ToeplitzMatrix(n, specL)

Parameters:

n	-	matrix order (symbolic or integer value)
specL	-	list of constant diagonals in the form: [[interval1, f1], [interval2, f2] , ...] where intervals are of the form p1..p2 (with p1<=p2) and must be a partition of [-n+1..n-1] [p1..p2, f] means that the diagonal i (p1<= i <= p2) is generated by the constant function f(i). (0=maindiagonal, 1= 1st upper sidediagonal, -1=1st lower side diagonal. etc.) Short cuts: f instead of [[1..n, f]] and [pos, f] instead of [pos..pos, f] .

Description:

- The function ToeplitzMatrix returns the specified Toeplitz matrix (a matrix with constant diagonals). If the order is symbolic then dots "o" are used to abbreviate the symbolic form; thus the returned matrix in this case should be used for illustrative purposes only, since Maple treats the "o"'s as usual matrix entries.
- This function is part of the SYMMETRIC package and so can be used in the form ToeplitzMatrix(..) only after setting the libname appropriately and performing the command with(SYMMETRIC) or with(SYMMETRIC,ToeplitzMatrix).

Examples:

```
>  with(SYMMETRIC):
>  ToeplitzMatrix(n,[[-n+1..-1,a],[0..1,b],[2..n -1,c]]);
```

$$\begin{bmatrix} b & b & c & c & o & o & c \\ a & b & b & c & c & o & o \\ a & a & b & b & c & c & o \\ o & a & a & b & b & c & c \\ o & o & a & a & b & b & c \\ a & o & o & a & a & b & b \\ a & a & o & o & a & a & b \end{bmatrix}$$

```
> ToeplitzMatrix(5,[[-4..-2,x[i]+1],[-1,f],[0.. 1,1],[2,0],[3..4,i]]);
```

$$\begin{bmatrix} 1 & 1 & 0 & 3 & 4 \\ f & 1 & 1 & 0 & 3 \\ x_{-2}+1 & f & 1 & 1 & 0 \\ x_{-3}+1 & x_{-2}+1 & f & 1 & 1 \\ x_{-4}+1 & x_{-3}+1 & x_{-2}+1 & f & 1 \end{bmatrix}$$

```
> ToeplitzMatrix(n,[[-n+1..-1,x],[0..3,y],[4..n -3,z],[n-2..n-1,a[i]]]);
```

$$\begin{bmatrix} y & y & y & y & z & z & o & o & z & a_{n-1} & a_n \\ x & y & y & y & y & z & z & o & o & z & a_{n-1} \\ x & x & y & y & y & y & z & z & o & o & z \\ x & x & x & y & y & y & y & z & z & o & o \\ x & x & x & x & y & y & y & y & z & z & o \\ o & x & x & x & x & y & y & y & y & z & z \\ o & o & x & x & x & x & y & y & y & y & z \\ x & o & o & x & x & x & x & y & y & y & y \\ x & x & o & o & x & x & x & x & y & y & y \\ x & x & x & o & o & x & x & x & x & y & y \\ x & x & x & x & o & o & x & x & x & x & y \end{bmatrix}$$

See Also: **ToeplitzDet**

Function: SYMMETRIC[ToeplitzDet] - determinant of specified Toeplitz matrix

Calling Sequence:

ToeplitzDet(n, specL)
ToeplitzDet(n, specL, print)
ToeplitzDet(n, specL, check)
ToeplitzDet(n, specL, print, check)

Parameters:

n	-	matrix order (symbolic or integer value)
specL	-	list of constant diagonals in the form: [[interval1, f1], [interval2, f2] , ...] where intervals are of the form p1..p2 (with p1<=p2) and must be a partition of [-n+1..n-1] [p1..p2, f] means that the diagonal i (p1<= i <= p2) is generated by the constant function f(i). (0=maindiagonal, 1= 1st upper sidediagonal, -1=1st lower side diagonal. etc.) Short cuts: f instead of [[1..n, f]] and [pos, f] instead of [pos..pos, f] .
print	-	optional display directive
check	-	optional checking directive
check[k]	-	optional checking directive (k positive integer)

Description:

- The function ToeplitzDet returns a determinant formula for the specified Toeplitz matrix (a matrix with constant diagonals). For symbolic orders this is only possible for some special cases, otherwise the unevaluated function call is returned.
- The optional directive check or check[k] , k positive integer, enables the checking mechanism: The determinant of a integer order matrix is computed using the standard det function and is compared with the value of the output formula for the same order. Default for the check order is 4 and can be set to an arbitrarily large value using check[k]. If the output formula is only valid for higher orders, the check order is automatically adapted. If the order n is an integer, the check value is set to n.
- The directive print (optional) prints the specified alternant matrix before returning the determinant formula.
- This function is part of the SYMMETRIC package and so can be used in the form ToeplitzDet(..) only after setting the libname appropriately and performing the command with(SYMMETRIC) or with(SYMMETRIC,ToeplitzDet).

Examples:

```
> with(SYMMETRIC):
> ToeplitzDet(n,[[-n+1..-3,a],[-2..3,b],[4..n-1 ,a]],print,check);
```

Matrix :

$$\begin{bmatrix} b & b & b & b & a & a & o & o & a \\ b & b & b & b & b & a & a & o & o \\ b & b & b & b & b & b & a & a & o \\ a & b & b & b & b & b & b & a & a \\ a & a & b & b & b & b & b & b & a \\ o & a & a & b & b & b & b & b & b \\ o & o & a & a & b & b & b & b & b \\ a & o & o & a & a & b & b & b & b \\ a & a & o & o & a & a & b & b & b \end{bmatrix}$$

Determinant :

$$formula\ valid\ for:,\ 8 \le n$$

$$[[(b-a)^{(n-1)}(b+k\_\,a),\ n=6\,k\_+1],\ [(b-a)^{(n-1)}(b+(k\_-1)\,a),\ n=6\,k\_],\ [0,\ otherwise]]$$

```
>  ToeplitzDet(n,[[-n+1..-1,c],[0,a],[1..n-3,b], [n-2..n-1,z[i]]],print,check);
```

$$Matrix:$$

$$\begin{bmatrix} a & b & b & o & o & b & z_{-2+n} & z_{n-1} \\ c & a & b & b & o & o & b & z_{-2+n} \\ c & c & a & b & b & o & o & b \\ c & c & c & a & b & b & o & o \\ o & c & c & c & a & b & b & o \\ o & o & c & c & c & a & b & b \\ c & o & o & c & c & c & a & b \\ c & c & o & o & c & c & c & a \end{bmatrix}$$

$$Determinant:$$

$$formula\ valid\ for:,\ 5 \le n$$

$$\begin{aligned} &a(\frac{(a-b)^n}{a-b} + 2\,\frac{(-1)^{(-3+n)}(-z_{-2+n}+b)(a-b)(c-a)^{(-2+n)}}{c-a} \\ &\quad + \frac{(-1)^{(-2+n)}(-z_{n-1}+z_{-2+n})(c-a)^{(n-1)}}{c-a}) \\ &\quad - b\,(c-a)\,(\frac{(a-b)^{(n-1)}}{a-b} + \frac{(-1)^{(-3+n)}(-z_{-2+n}+b)(c-a)^{(-2+n)}}{c-a}) \\ &\quad + \frac{b\,(a-b)^{(n-1)}\,(\frac{-c+a}{a-b})^{(-2+n)}(a-b)}{-c+b} - \frac{b\,(a-b)^{(n-1)}(-c+a)^2}{(a-b)(-c+b)} \\ &\quad + (-1)^{(-2+n)}(c-a)^{(-2+n)}z_{-2+n}(a-b) + (-1)^{(n-1)}(c-a)^{(n-1)}z_{n-1} \end{aligned}$$

```
>  ToeplitzDet(n,[[-n+1..-1,b],[0..1,a[i]],[2..n -2,b],[n-1,z]],print,check);
```

$$Matrix:$$

$$\begin{bmatrix} a_0 & a_1 & b & b & o & o & b & z \\ b & a_0 & a_1 & b & b & o & o & b \\ b & b & a_0 & a_1 & b & b & o & o \\ b & b & b & a_0 & a_1 & b & b & o \\ o & b & b & b & a_0 & a_1 & b & b \\ o & o & b & b & b & a_0 & a_1 & b \\ b & o & o & b & b & b & a_0 & a_1 \\ b & b & o & o & b & b & b & a_0 \end{bmatrix}$$

$$Determinant:$$

$$formula\ valid\ for:,\ 5 \le n$$

$$n \ >= \ 2$$

$$\frac{a_0\,(a_0-b)^{(1+n)}}{(a_0-b)^2} - \frac{b\,(a_0-b)^n\,(a_1-a_0)}{(a_0-b)^2} - (-1)^n\,b(2\,b\,(b-a_0)^n\,n - 2\,b\,(b-a_0)^n + (a_1-b)^n\,b$$
$$- a_0\,(b-a_0)^n\,n + (b-a_0)^n\,a_1 - (a_1-b)^n\,a_0 + a_0\,(b-a_0)^n - (b-a_0)^n\,n\,a_1)\Big/(-a_1+2\,b-a_0)^2$$
$$- \frac{(a_0-b)^n\,b\,(4\,b\,(b-a_0)^3 + (a_1-b)^3\,b - 2\,a_0\,(b-a_0)^3 - 2\,(b-a_0)^3\,a_1 - (a_1-b)^3\,a_0)}{(-a_1+2\,b-a_0)^2\,(a_0-b)^3} - (-1)^{(1+n)}$$
$$b(b^2\,(b-a_0)^{(-2+n)} + b\,(b-a_0)^{(-2+n)}\,a_0 - 2\,b\,(b-a_0)^{(-2+n)}\,z - b\,(b-a_0)^{(-2+n)}\,a_1$$
$$- (b-a_0)^{(-2+n)}\,{a_0}^2 + (b-a_0)^{(-2+n)}\,z\,a_1 + (a_1-b)^n + (b-a_0)^{(-2+n)}\,z\,a_0)\Big/($$
$$-a_1+2\,b-a_0)$$

See Also: **ToeplitzMatrix**

Bibliography

[ABD$^+$97] F. Avnaim, J. Boissonat, O. Devillers, F. Preparata, and M. Yvinec. Evaluating signs of determinants using single–precision arithmetic. *Algorithmica*, 17:111–132, 1997.

[Bar67] E.H. Bareiss. Sylvesters identity and multistep integer preserving gaussian elimination. *Math. Comput.*, 8:565–578, 1967.

[Bar68] E.H. Bareiss. Computational solution of matrix problems over an integral domain. *J. Inst. Maths. Applics.*, 10:68–104, 1968.

[Bri83] E. Brieskorn. *Lineare Algebra und analytische Geometrie I.* Vieweg, 1983.

[BS91] I. Bronstein and K. Semendjajew. *Taschenbuch der Mathematik.* Teubner, 25th edition, 1991.

[BY96] H. Brönnimann and M. Yvinec. A complete analysis of Clarkson's algorithm for safe determinant evaluation. Technical Report 3051, INRIA, 1996.

[BY97] H. Brönnimann and M. Yvinec. Efficient exact evaluation of signs of determinants. In *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, pages 166–173, 1997.

[CGG$^+$92] B. Char, K. Geddes, G. Gonnet, B. Leong, M. Monagan, and S. Watt. *Maple V Language Reference Manual*, 1992.

[CGGG83] B. Char, K. Geddes, W. Gentleman, and G. Gonnet. The design of Maple : A compact, portable and powerful computer algebra system. In *Proceedings of Eurocal '83*, 1983.

[Cla92] K. Clarkson. Safe and effiecient determinant evaluation. In *Proc. 33rd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 387–395, 1992.

[CLO96] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties and Algorithms.* Springer, second edition, 1996.

[DW95] A. Dress and W. Wenzel. A simple proof of an identity concerning pfaffians of skew symmetric matrices. *Advances in Mathematics*, 112:120–134, 1995.

[ES95] J. Erickson and R. Seidel. Better lower bounds on detecting affine and spherical degeneracies. *Discrete Computational Geometry*, 13:41–57, 1995.

[FW93] S. Fortune and C. Van Wyk. Efficient exact arithmetic for computational geometry. In *Proc. 9th Ann. Symp. Comp. Geom.*, pages 163–172, 1993.

[GKP92] R. Graham, D. Knuth, and O. Patashnik. *Concrete Mathematics.* Addison–Wesley, 1992.

[Gol91] D. Goldberg. What every computer scientist should know about floating–point arithmetic. *ACM Computing Surveys*, 21(1):5–48, 1991.

[GvL96] G. Golub and C. van Loan. *Matrix Computations.* Johns Hopkins University Press, third edition, 1996.

[HS75] E. Horowitz and S. Sahni. On computing the exact determinant of matrices with polynomial entries. *Journal of the ACM*, 22:38–50, 1975.

[Jän91] K. Jänich. *Lineare Algebra.* Springer, 1991.

[Kal] K. Kalorkoti. Introduction to computer algebra. Edinburgh CS4 Course Notes.

[Knu96] D. Knuth. Overlapping pfaffians. *Electronic Journal of Combinatorics*, 3, 1996. available at `http://www-cs-faculty.stanford.edu/~knuth/musings.html`.

[LP86] L. Lovász and M. Plummer. *Matching Theory.* Number 29 in Annals of Discrete Mathematics. Elsevier Science Publishers, 1986.

[Met60] W. Metzler. *A Treatise on the Theory of Determinants by Thomas Muir.* Dover reprint, 1960.

[Mit81] O. Mitchell. Note on the determinant of powers. *American Journal of Mathematics*, 4:341–344, 1881.

[Mon] M. Monagan. *Programming in Maple: The Basics.*

[Red94] D. Redfern. *Maple Handbook, Maple V R3*, 1994.

[SB90] J. Stoer and R. Bulirsch. *Numerische Mathematik II.* Springer, third edition, 1990.

[SM82] T. Sasaki and H. Murao. Efficient gaussian elimination method for symbolic determinants and linear systems. *ACM Trans. Math. Software*, 8:277–289, 1982.

[Sto94] J. Stoer. *Numerische Mathematik I.* Springer, 7th edition, 1994.

[SW89] U. Storch and H. Wiebe. *Lehrbuch der Mathematik: Band II: Lineare Algebra.* BI Wissenschaftsverlag, 1989.

[Val79] L. Valiant. Completeness classes in algebra. In *Proc. 11th Ann. Symp. Theory Comput.*, pages 249–261, 1979.