A Rigorous Proofless Approach to Linear Algebra

Murisi Tarusenga

Sunday 22$^{\text{nd}}$ October, 2017 16:27

1 Introduction

Only first 14 pages have been revised. Think of last 14 pages as a roadmap.

What follows is an attempt to present linear algebra as a system of algorithms in a manner devoid of argumentation. The correctness of these algorithms hopefully is made apparent by stating expected variable values at certain points of execution. Mathematical argumentation is circumvented by letting algorithms *show* what a proposition in an argument might have *said*.

The subset of linear algebra I intend to tackle consists of the Smith normal form, determinants, compound matrices, general solutions to linear systems, characteristic matrices, the rational canonical form, block matrix multiplication, minimum polynomials, orthogonalization, and the spectral theorem for symmetric matrices. To get the above done, I also have to extract the algorithms from Sturm's theorem, Cauchy's bound, the Cauchy-Schwarz inequality, the Euclidean division algorithm, and the factor theorem.

Some things to take into consideration:

1. Where a proof expresses generality using universal quantification, the algorithms below show generality by allowing one to choose inputs. See algorithm 24.

2. Where a proof expresses that a proposition implies contradiction, the algorithms below merely show how certain control flows lead to absurdity. See algorithms 11, 28, 34, and 51.

3. Where a proof uses the principle of mathematical induction, the algorithms below merely use iteration/recursion and loop invariants. See algorithms 46, 46 auxilliary, and 51.

Anyway, to see most easily what I am trying to do, see algorithms 33, 39, 48, and 50.

Contents

2 Body

2.1 Algorithm 1

Choose a 1×2 matrix, A, whose entries are polynomials and do the following:

1. Let A be our working matrix.
2. If the $A_{1,1} = 0$, then add $A_{1,2}$ to it.
3. Then while $A_{1,2} \neq 0$, do the following:
   (a) If $\deg(A_{1,1}) = \deg(A_{1,2})$ and $A_{1,1}$ is monic, then:
      i. Subtract a times $A_{1,1}$ from $A_{1,2}$ where a is the leading coefficient of $A_{1,2}$.
   (b) Otherwise, if $\deg(A_{1,1}) = \deg(A_{1,2})$ but $A_{1,1}$ is not monic, then:
      i. Add $1 - b/a$ times $A_{1,2}$ to $A_{1,1}$ where b and a are the leading coefficients of $A_{1,1}$ and $A_{1,2}$ respectively.
      ii. Go to (3a) again.
   (c) Otherwise, if $\deg(A_{1,1}) \neq \deg(A_{1,2})$, then:
      i. Let p and q be locations of the polynomials with the lower and higher degree respectively.
      ii. Add $-b/ax^{\deg(p)-\deg(q)}$ times p to q where b and a are the leading coefficients of q and p respectively.
      iii. If q becomes zero, then:
         A. Undo operation (3cii) and do the same thing again, only this time using the fraction $1 - b/ax^{\deg(p)-\deg(q)}$.
   (d) **Verify that the degree of only one entry changed.**
   (e) **Verify that the changed entry's degree decreased.**
4. **Verify that $A_{1,2} = 0$.**
5. If the original A was not zero, then do the following:
   (a) Verify that the body of loop (3) executed at least once.
   (b) Verify that the last instructions executed were an instance of (3ai), (3d), then (3e).
   (c) **Therefore verify that $A_{1,1}$ is a monic polynomial.**
6. Otherwise, do the following:
   (a) Verify that both entries were originally zero.
   (b) **Therefore, now verify that $A_{1,1} = 0$.**
7. **Yield the tuple $\langle A \rangle$.**

2.2 Algorithm 2

Choose an $m \times n$ matrix, A, whose entries are polynomials and do the following:

1. Let A be our working matrix.
2. If the top-left entry of the matrix is zero, then do the following:
   (a) While there are non-zero entries in the top row less its first entry, do the following:
      i. In the first row, select the 1×2 matrix whose right entry coincides with the last non-zero entry of the first row
      ii. Apply algorithm 1 on this submatrix but this time adding and subtracting the entire columns instead of merely just the entries in the submatrix.
      iii. Verify that the left and right entries of the submatrix are now non-zero and zero respectively.
      iv. If the left entry of the submatrix coincides with the top-left entry of the matrix,
         A. Verify that the top-left entry is now non-zero.
         B. Go to operation (2).
   (b) Now do the same operations as in (a), but this time with the operations themselves reflected across the matrix's diagonal. I.e. making 2×1 submatrices starting from the bottom-most non-zero row of the first column and working your way upwards.
   (c) Verify that, except for the top-left entry, the first row and the first column are zero. Now skip to operation (3).
3. While the top-left entry is non-zero, do the following:
   (a) Call operation (1a) except that at (1aivA), we instead expect that the top-left entry has a lower degree than before.
   (b) Call operation (1b) except that at (1bivA), we instead expect that the top-left entry has a lower degree than before.
   (c) Call operation (1c).
4. Apply algorithm 2 to the $(m-1) \times (n-1)$ submatrix formed by removing the first row and first column from the matrix under consideration.
5. Verify that (3)'s execution leaves the first row and column unchanged.
6. **Verify that A is now a diagonal matrix.**
7. **Yield the tuple $\langle A \rangle$.**

2.3 Algorithm 3 (Smith normal form construction)

Choose an $m \times n$ matrix, A, whose entries are polynomials and do the following:

1. Apply algorithm 2 on A.
2. If $m > 0$ and $n > 0$, then do the following:
   (a) For j going from 2 to $\min(m, n)$, do the following:
      i. Add row j to row 1 and let A', a copy of A, be our working matrix.
      ii. Apply algorithm 1 on the submatrix of A' formed by selecting row 1 and columns 1 and j as if there were nothing in between.
      iii. Verify that the execution of algorithm 1 in (ii) manifested itself in a sequence of column operations to make $A'_{1,j}$ zero.
      iv. Temporarily stepwise undo these column operations and do the following:
         A. Verify that $A_{1,1} = p * A'_{1,1}$, where p is some implicitly constructed polynomial.
         B. Verify that $A_{1,1}$ is a factor of all $A_{2,2}, \cdots, A_{j-1,j-1}$.
         C. Therefore verify that $A'_{1,1}$ is also a factor of all $A_{2,2}, \cdots, A_{j-1,j-1}$.
         D. Verify that $A_{j,j} = A_{1,j} = q * A'_{1,1}$, where q is some implicitly constructed polynomial.
         E. Verify that $A'_{j,1} = r * A_{j,j} = r * A_{1,j} = rq * A'_{1,1}$, where r is some implicitly constructed polynomial.

F. Verify that $A'_{j,j} = t * A_{j,j} = t * A_{1,j} = tq * A'_{1,1}$, where t is some implicitly constructed polynomial.

v. Subtract rq times row 1 from row j.

vi. Now verify that $A'_{j,1} = 0$.

vii. Now let A be equal to our working matrix.

(b) Call (2) in-place on the submatrix formed by removing the first row and column.

(c) Verify that each entry on the diagonal after $A_{1,1}$ is a linear combination of multiples of $A_{1,1}$.

(d) Therefore verify that each entry on the diagonal is still be a multiple of $A_{1,1}$.

(e) **Let $A_{0,0} = 1$.**

(f) **Verify that for all $1 \leq i \leq \min(m, n)$, $A_{i,i} = u_i A_{i-1,i-1}$, where u_i is a polynomial implicitly constructed above.**

3. **Yield the tuple $\langle A \rangle$.**

2.4 Algorithm 4 (Associativity verification)

Choose an $m \times n$ matrix, A, an $n \times p$ matrix, B, and a $p \times q$ matrix C, all of whose entries are polynomials. Now do the following:

1. Verify that $(AB)_{i,l} = \sum_{k=0}^{n-1} (A_{i,k} * B_{k,l})$.

2. Verify that $((AB)C)_{i,r} = \sum_{l=0}^{p-1} ((AB)_{i,l} * C_{l,r}) = \sum_{l=0}^{p-1} \left(\sum_{k=0}^{n-1} (A_{i,k} * B_{k,l}) * C_{l,r} \right)$.

3. Verify that $(BC)_{k,r} = \sum_{l=0}^{p-1} (B_{k,l} * C_{l,r})$.

4. Verify that $(A(BC))_{i,r} = \sum_{k=0}^{n-1} (A_{i,k} * (BC)_{k,r}) = \sum_{k=0}^{n-1} \left(A_{i,k} * \sum_{l=0}^{p-1} (B_{k,l} * C_{l,r}) \right)$.

5. Verify that (2) $= \sum_{l=0}^{p-1} \left(\sum_{k=0}^{n-1} (A_{i,k} * B_{k,l} * C_{l,r}) \right) = \sum_{k=0}^{n-1} \left(\sum_{l=0}^{p-1} (A_{i,k} * B_{k,l} * C_{l,r}) \right) = \sum_{k=0}^{n-1} \left(A_{i,k} * \sum_{l=0}^{p-1} (B_{k,l} * C_{l,r}) \right) = (4)$.

6. **Therefore verify that $(AB)C = A(BC)$.**

2.5 Algorithm 5 (Extended Smith normal form construction)

Choose an $m \times n$ matrix, A, whose entries are polynomials and do the following:

1. Make a singleton list containing one item, the chosen matrix A.

2. Augment algorithm 3 so that each time a polynomial x times a column i is added onto column j, an $n \times n$ matrix that only has 1s on its diagonal, and such that the only non-zero entry off its diagonal is x at position (i, j), is appended onto the list.

3. Also augment algorithm 3 so that each time a polynomial x times a row i is added onto row j, an $n \times n$ matrix that only has 1s on its diagonal, and such that the only non-zero entry off its diagonal is x at position (j, i), is prepended onto the list.

4. Now run algorithm 3 on the matrix A.

5. Let D be the diagonal matrix produced by algorithm 3.

6. **Verify that D equals the product, in-order, of the matrices in the list.**

7. Let M be the product of the sublist whose entries are all those that precede A.

8. Let N be the product of the sublist whose entries are all those that follow the original A.

9. **Verify that $D = MAN$.**

10. **Yield the tuple $\langle D, M, N \rangle$.**

2.6 Algorithm 6 (Inverse extended Smith normal form construction)

Choose an $m \times n$ matrix, A, whose entries are polynomials and do the following:

1. Run algorithm 3 on matrix A to get the matrix D.

2. Now, starting work with the matrix D, iterate through the row and column operations of algorithm 3 starting from the last and ending with the first:

   (a) If the current operation adds polynomial x times column i to column j, then add

$-x$ times column i from column j from our working matrix.

   (b) If the current operation adds polynomial x times row i to row j, then add $-x$ times row i from row j from our working matrix.

3. Our working matrix should now be equal to A.
4. Make a singleton list containing one item, the original matrix D.
5. Augment (2) in a way analogous to how algorithm 5 augmented algorithm 3, but this time with the list provided in (4).
6. Now run augmented (2).
7. **Verify that A equals the product, in-order, of the matrices in the list.**
8. Let M^{-1} be the product of the sublist whose entries are all those that precede D.
9. Let N^{-1} be the product of the sublist whose entries are all those that follow D.
10. **Verify that $A = M^{-1}DN^{-1}$.**
11. **Yield the tuple $\langle M^{-1}, D, N^{-1}\rangle$.**

2.7 Algorithm 7

Choose an $m \times n$ matrix, A, whose entries are polynomials and do the following:

1. Run algorithm 6 with A as the choice matrix.
2. Now, starting work with the matrix I_n, the $n \times n$ matrix with only 1s on the diagonal, do the following operations:
   (a) Apply in-order the column operations that were applied on A
   (b) Then apply in-order the column operations that were applied on D
3. Verify that the application of operation (2) leaves I_n unchanged.
4. Verify that $(I_nN)N^{-1}$ evaluates to the same matrix produced by (2).
5. **Therefore verify that $(I_nN)N^{-1} = I_n$.**
6. **Therefore, using algorithm 4, verify that $I_n = (I_nN)N^{-1} = I_n(NN^{-1}) = NN^{-1}$.**

Using similar computations, verify that $N^{-1}N = I_n$, and that $MM^{-1} = M^{-1}M = I_m$.

2.8 Algorithm 8 (Determinant calculation)

Choose an $m \times m$ matrix, A, whose entries are polynomials and do the following:

1. If m equals 0, then do the following:
   (a) **Yield the tuple $\langle 1\rangle$.**
2. Otherwise, do the following:
   (a) Let a be the sum of m terms where, counting from $i = 0$, the i^{th} term is $((-1)^i A_{i,1}$ times the result of applying algorithm 8 on the submatrix formed by removing the first column and i^{th} row from A).
   (b) **Yield the tuple $\langle a\rangle$.**

We will use the notation $\det_{I,J}(A)$ to refer to the invocation of algorithm 8 on the square submatrix created by selecting the sequence of rows I and the sequence of columns J from A. If neither I nor J are specified, then the invocation shall happen directly on A.

2.9 Algorithm 9 (Multilinearity verification)

Choose a polynomial p. Choose two $m \times 1$ matrices, B and C, whose entries are polynomials. Choose an $m \times m$ matrix, A, whose entries are polynomials and that is such that its i^{th} column is $B + pC$. The value $\det(A)$ can be evaluated as follows:

1. Considering every element of A to be a unit, verify that fully expanding out $\det(A)$ yields an alternating sum of $m!$ terms, where each term is the product of m entries of A, where each entry has a distinct row and distinct column.
2. Distribute out the entries of the i^{th} column occuring in this alternating sum.
3. Verify that the outcome of (2) is $m! * 2$ terms.
4. Reorder the terms so that the currently odd ones come first and the currently even ones come last.

5. Verify that $\det(A) = \det(A') + \det(A'')$ where A' is A with the i^{th} column replaced by B and A'' is A with the i^{th} column replaced by pC.

6. Reorder the factors of each term in $\det(A'')$ to bring p to the front.

7. Now verify that the $\det(A'') = p\det(A''')$, where A''' is A with the i^{th} column replaced by C.

8. **Therefore verify that** $\det(A) = \det(A') + p\det(A''')$**.**

Make an analogous algorithm for cases when a given row is the sum of two $1 \times m$ matrices.

2.10 Algorithm 10 (Alternation verification)

Choose an $m \times m$ matrix, A, whose entries are polynomials. Choose a row $1 < i \leq m$. To evaluate $\det(A')$ where A' is A with rows $i-1$ and i swapped, do the following:

1. Fully expand out $\det A$ into $m!$ terms and then do the same for $\det A'$.

2. For each of the $m!$ ways to select m rows from A, let $r = (r_1, r_2, \cdots, r_m)$ be the rows selected corresponding to the columns $1, 2, \cdots, m$ respectively, and do the following:

   (a) Verify that the values selected by r in A are the same as the values selected by r' in A', where r' is obtained by swapping the values $i-1$ and i in the sequence r.

   (b) Execute algorithm 8 on A, and consider the execution path that produces the term corresponding to the row selections r. Ditto for A' and r'.

   (c) Let k be the lesser of the indicies of the values $i-1$ and i in the sequence r.

   (d) Verify that the signs attached to $A_{r_1,1}, \cdots, A_{r_{k-1},k-1}$ are the same as the signs attached to $A'_{r'_1,1}, \cdots, A'_{r'_{k-1},k-1}$.

   (e) Verify that indices r_k and r'_k identify adjacent rows in the remaining respective submatrices of A and A'.

   (f) Therefore verify that the signs then attached to A_{r_k} and $A'_{r'_k}$ are opposite.

   (g) Verify that after the removal of r_k and r'_k from their respective submatrices, the submatrices left are identical.

   (h) Therefore verify that the signs attached to $A_{r_{k+1},k+1}, \cdots, A_{r_{m-1},m-1}$ are the same as the signs attached to $A'_{r'_{k+1},k+1}, \cdots, A'_{r'_{m-1},m-1}$.

   (i) Therefore verify that the term corresponding to the row selections r' in the full expansion of $\det(A')$ has the opposite sign to the term corresponding to the row selections r in the full expansion of $\det(A)$.

3. Therefore verify that every term in the full expansion of $\det(A)$ corresponds to a unique negated version of itself in the full expansion of $\det(A')$.

4. **Therefore verify that** $\det(A') = -\det(A)$**.**

Make a simpler algorithm to verify that column swaps cause sign alternations.

2.11 Algorithm 11

Choose integers $1 < i \leq m$. Choose an $m \times m$ matrix, A, whose entries are polynomials, and such that columns $i-1$ and i are the same. To evaluate $\det(A)$, do the following:

1. If $\det(A) \neq 0$, then do the following:

   (a) Let A' be A with columns i and $i-1$ swapped.

   (b) Verify that A' equals A.

   (c) Therefore verify that $\det(A') = \det(A)$.

   (d) Using algorithm 10, also verify that $\det(A') = -\det(A)$.

   (e) **Abort algorithm.**

2. Otherwise, do the following:

   (a) **Verify that** $\det(A) = 0$**.**

Make an analogous algorithm to verify that matrix choices with repeated rows yield determinants equal to zero.

2.12 Algorithm 12

Choose a column index $1 \leq i \leq m$. Choose a positive integer j. Choose an $m \times m$ matrix, A, whose entries are polynomials. To evaluate $\det(A')$ where A' is A but with column i moved j places, do the following:

1. Let $A_i = A$.
2. For $k = i + 1$ to $k = i + j$, do the following:
   (a) Let A_k be obtained by swapping columns $k - 1$ and k of A_{k-1}.
   (b) Using algorithm 10, verify that $\det(A_k) = -\det(A_{k-1})$.
3. Verify that $A' = A_{i+j}$.
4. **Therefore verify that** $\det(A') = \det(A_{i+j}) = (-1)^1 \det(A_{i+j-1}) = \cdots = (-1)^j \det(A_i) = (-1)^j \det(A)$**.**

Make an analogous algorithm that verifies that $\det(A') = (-1)^j \det(A)$ when a non-positive integer, j, is chosen.

Also make an analogous algorithm that does the verification for moved rows.

2.13 Algorithm 13 (Compound matrix calculation)

Choose an $m \times n$ matrix, A, of polynomials and choose an integer $1 \leq k \leq \min(m, n)$. Yield a tuple comprising the $\binom{m}{k} \times \binom{n}{k}$ matrix constructed as follows:

1. The rows are labeled by the colexicographically sorted list of increasing length-k sequences whose elements are picked from the first m positive integers.
2. The columns are labeled by the colexicographically sorted list of increasing length-k sequences whose elements are picked from the first n positive integers.
3. For each row label I: For each column label J: Let the entry at position (I, J) be $\det_{I,J}(A)$.

We will use the notation $C_k(A)$ to refer to an invocation of algorithm 13 on the matrix A.

2.14 Algorithm 14 (Compound matrix of identity calculation)

Choose two integers $0 \leq k \leq m$. To evaluate $C_k(I_m)$, iterate through all its entries and do the following:

1. Algorithm 13 requires us to form a submatrix B of A using the rows listed in the row label, and the columns listed in the column label.
2. If the entry is diagonal, then do the following:
   (a) Verify that the k column indices we just selected from I_m are the same as the k rows indices we just selected from I_m.
   (b) Therefore verify that the diagonal positions of B correspond to a diagonal entry of I_m.
   (c) Therefore verify that the diagonal positions of B are 1.
   (d) Also verify that the non-diagonal positions of B correspond to a non-diagonal entry of I_m.
   (e) Therefore verify that the non-diagonal positions of B are 0.
   (f) Therefore the submatrix B should equal I_k.
   (g) **Therefore using algorithm 8, verify that $|B| = 1$.**
3. Otherwise, do the following:
   (a) Verify that the k column indices we just selected from I_m are different from the k row indices that we just selected from I_m.
   (b) Let i be a selected row index that is not also a column index.
   (c) Iterate through the columns of the row in the submatrix B that corresponds to row i of I_m and do the following:
      i. Let j be the column index of I_m to which this column corresponds.
      ii. Using (3b), verify that $i \neq j$.
      iii. Verify that $(I_m)_{i,j} = 0$.
      iv. Therefore verify that this entry is 0.
   (d) Verify that the row in the submatrix B that corresponds to row i of I_m is entirely zero.

(e) **Therefore using algorithm 8, verify that $|B| = 0$.**

4. **Therefore verify that $C_k(I_m) = I_{\binom{m}{k}}$.**

2.15 Algorithm 15

Choose an integer $1 \leq k \leq \min(m, n)$. Choose an $m \times m$ matrix, A, whose diagonal entries are 1s, and such that the only entry off the diagonal is the polynomial p at (i, j). Also choose an $m \times n$ matrix, B, whose entries are polynomials. To evaluate $C_k(AB)$, do the following:

1. Verify that AB equals B, but with its row i having p times B's row j added to it.
2. Go through the row labels, I, of $C_k(AB)$ and do the following:
   (a) If $i \notin I$, then do the following:
      i. For $l \in I$: For $j = 1$ to $j = n$: Verify that $(AB)_{l,j} = B_{l,j}$.
      ii. Therefore for each column label J, verify that $C_k(AB)_{I,J} = \det_{I,J}(AB) = \det_{I,J}(B) = C_k(B)_{I,J}$.
      iii. **Therefore verify that row I of $C_k(AB)$ equals row I of $C_k(B)$.**
   (b) Otherwise, if $i \in I$, then:
      i. Let I' be I but with an in-place replacement of i by j.
      ii. For each column label J: Using algorithm 9, verify that $C_k(AB)_{I,J} = \det_{I,J}(AB) = \det_{I,J}(B) + p * \det_{I',J}(B)$.
      iii. Let l be the signed number of places that the j introduced above needs to be moved in order to make I' a non-increasing sequence.
      iv. Let I'' be obtained from I' by moving the integer j in I' by l places.
      v. For each column label J: Using algorithm 12, verify that $\det_{I',J}(B) = (-1)^l \det_{I'',J}(B)$.
      vi. Therefore for each column label J: Verify that $C_k(AB)_{I,J} = \det_{I,J}(B) + p * \det_{I',J}(B) = \det_{I,J}(B) + (-1)^l p * \det_{I'',J}(B)$.
      vii. If $j \notin I$, then do the following:
         A. Verify that I'' is a decreasing sequence.
         B. Verify that I'' is a row label of $C_k(B)$.
         C. Therefore for each column label J: Verify that $C_k(AB)_{I,J} = \det_{I,J}(B) + (-1)^l p * \det_{I'',J}(B) = C_k(B)_{I,J} + (-1)^l p * C_k(B)_{I'',J}$.
         D. **Therefore verify that row I of $C_k(AB)$ is row I of $C_k(B)$ plus $(-1)^l p$ times row I'' added to it.**
      viii. Otherwise if $j \in I$, do the following:
         A. Verify that the sequence I'' contains two consecutive js.
         B. Using algorithm 11, verify that $\det_{I'',J}(B) = 0$.
         C. Therefore verify that $C_k(AB)_{I,J} = \det_{I,J}(B) = C_k(B)_{I,J}$.
         D. **Therefore verify that row I of $C_k(AB)$ is row I of $C_k(B)$.**
3. Let D be the matrix of row operations implicitly applied in (2).
4. **Verify that $C_k(AB) = DC_k(B)$.**

2.16 Algorithm 16

Choose an $m \times n$ matrix, A, whose only entries are polynomials on the diagonal positions. Also choose an $n \times n$ matrix, B, whose entries are polynomials. Also choose an integer $0 \leq k \leq \min(m, n)$. To evaluate $C_k(AB)$, do the following:

1. Verify that AB equals the top $m \times n$ submatrix of B with each row i multiplied by $A_{i,i}$.
2. Go through the row labels, I, of $C_k(AB)$ and do the following:
   (a) Let $(r_1, r_2, \cdots, r_k) = I$.
   (b) If $r_k \leq n$, then do the following:

i. Using algorithm 13, verify that every element of I is less than or equal to n.

ii. Let $A_0 = A$.

iii. For $i = 1$ to $i = k$: Let A_i equal A_{i-1} but with position (r_i, r_i) set to 1.

iv. For each column label J: Repeatedly using algorithm 9, verify that $C_k(AB)_{I,J} = \det_{I,J}(AB) = \det_{I,J}(A_0B) = A_{r_1,r_1}\det_{I,J}(A_1B) = A_{r_1,r_1}A_{r_2,r_2}\det_{I,J}(A_2B) = \cdots = A_{r_1,r_1}A_{r_2,r_2}\cdots A_{r_k,r_k}\det_{I,J}(A_kB) = A_{r_1,r_1}A_{r_2,r_2}\cdots A_{r_k,r_k}\det_{I,J}(B) = A_{r_1,r_1}A_{r_2,r_2}\cdots A_{r_k,r_k}C_k(B)_{I,J}$.

v. **Therefore verify that row I of $C_k(AB)$ is $A_{r_1,r_1}A_{r_1,r_1}\cdots A_{r_k,r_k}$ times row I of $C_k(B)$.**

(c) Otherwise if $r_k > n$, then do the following:

i. Verify that row r_k of A is zero.

ii. Therefore verify that row r_k of AB is zero.

iii. Therefore verify that the last row of the submatrix obtained by selecting rows I from AB is zero.

iv. Therefore using algorithm 8, for each column label J: verify that $C_k(AB)_{I,J} = 0$.

v. **Therefore verify that row I of $C_k(AB)$ is zero.**

3. Let D be the matrix of row operations implicitly applied in (2).

4. **Verify that D is diagonal.**

5. **Verify that $C_k(AB) = DC_k(B)$.**

2.17 Algorithm 17

Choose an integer $1 \leq k \leq \min(m,n)$. Choose an $m \times m$ matrix, A, whose diagonal entries are 1s, and such that the only entry off the diagonal is the polynomial p at (i,j). Also choose an $m \times n$ matrix, B, whose entries are polynomials. To evaluate $C_k(AB)$, do the following:

1. Execute algorithm 15 on matrices A and I_m. Let D be the matrix constructed.

2. Verify that $C_k(AI_m) = DC_k(I_m)$.

3. Using algorithm 14, verify that $C_k(A) = C_k(AI_m) = DC_k(I_m) = DI_{\binom{m}{k}} = D$.

4. Execute algorithm 15 on matrices A and B. Let D' be the matrix constructed.

5. Verify that $C_k(AB) = D'C_k(B)$.

6. Verify that $D' = D = C_k(A)$.

7. **Therefore verify that $C_k(AB) = C_k(A)C_k(B)$.**

Make an analogous algorithm to verify that $C_k(BA) = C_k(B)C_k(A)$.

Using algorithm 16, make an algorithm similar to above but that works when a diagonal matrix of polynomials, A, is instead chosen.

2.18 Algorithm 18 (Compound matrix of matrix product calculation)

Choose an integer $0 \leq k \leq \min(m,n,p)$. Choose an $m \times n$ matrix, A, whose only entries are polynomials. Also choose an $n \times p$ matrix, B, whose entries are polynomials. To evaluate $C_k(AB)$, do the following:

1. Execute algorithm 6 on A. Let ${M^{-1}}_i$ be the i^{th} element of the sublist corresponding to M and ${N^{-1}}_i$ be the i^{th} element of the sublist corresponding to N.

2. Verify that $A = {M^{-1}}_1 \cdots {M^{-1}}_m D {N^{-1}}_1 \cdots {N^{-1}}_n$.

3. **Using repeated applications of algorithm 17, verify that $C_k(AB) = C_k({M^{-1}}_1 \cdots {M^{-1}}_m D {N^{-1}}_1 \cdots {N^{-1}}_n B) = C_k({M^{-1}}_1) \cdots C_k({M^{-1}}_m) * C_k(D) * C_k({N^{-1}}_1) \cdots C_k({N^{-1}}_n) C_k(B) = C_k({M^{-1}}_1 \cdots {M^{-1}}_m D {N^{-1}}_1 \cdots {N^{-1}}_n) C_k(B) = C_k(A)C_k(B)$.**

2.19 Algorithm 19 (Determinant equals product of diagonal entries verification)

Choose an $m \times m$ matrix, A, whose entries are polynomials. To evaluate $\det(A)$, do the following:

1. Execute algorithm 6 on A. Let ${M^{-1}}_i$ be the i^{th} element of the sublist corresponding to M and ${N^{-1}}_i$ be the i^{th} element of the sublist corresponding to N.
2. Verify that $A = {M^{-1}}_1 \cdots {M^{-1}}_m D {N^{-1}}_1 \cdots {N^{-1}}_n$.
3. **Using algorithm 8 and algorithm 18, verify that $\det(A) = C_m(A) = C_m({M^{-1}}_0 \cdots {M^{-1}}_m D {N^{-1}}_0 \cdots {N^{-1}}_n) = C_m({M^{-1}}_0) \cdots C_m({M^{-1}}_m) C_m(D) C_m({N^{-1}}_0) \cdots C_m({N^{-1}}_n) = 1 \cdots 1 C_m(D) 1 \cdots 1 = C_m(D) = \det(D)$.**
4. **Using algorithm 8, verify that $\det(D)$ is the product of the diagonal entries of D.**

2.20 Algorithm 20 (Transpose calculation)

Choose an $m \times n$ matrix, A, whose entries are polynomials and do the following:

1. Make an $n \times m$ matrix, A^T.
2. For $i = 1$ to $i = n$:
   (a) For $j = 1$ to $j = m$:
      i. Let ${A^T}_{i,j} = A_{j,i}$.
3. **Yield the tuple $\langle A^T \rangle$.**

The notation A^T shall be used to refer to the result of invoking algorithm 20 on a matrix A.

2.21 Algorithm 21 (Transpose of product verification)

Choose an $m \times n$ matrix, A, and an $n \times k$ matrix, B, both of whose entries are polynomials. Now do the following:

1. Verify that $B^T A^T$ and $(AB)^T$ have dimensions $k \times m$.
2. For $i = 1$ to $i = k$:
   (a) For $j = 1$ to $j = m$:
      i. Using algorithm 20, verify that $(B^T A^T)_{i,j} = \sum_{l=0}^{n} B_{l,i} A_{j,l} = \sum_{l=0}^{n} A_{j,l} B_{l,i} = (AB)_{j,i} = ((AB)^T)_{i,j}$.
3. **Therefore verify that $B^T A^T = (AB)^T$.**

2.22 Algorithm 22 (Determinant of transpose verification)

Choose an $m \times m$ matrix, A, whose entries are polynomials. To evaluate $\det(A^T)$, do the following:

1. Execute algorithm 6 on A. Let ${M^{-1}}_i$ be the i^{th} element of the sublist corresponding to M and ${N^{-1}}_i$ be the i^{th} element of the sublist corresponding to N.
2. Verify that $A = {M^{-1}}_1 \cdots {M^{-1}}_m D {N^{-1}}_1 \cdots {N^{-1}}_n$.
3. **Therefore using algorithms 19 and 20, verify that $\det(A^T) = \det(({M^{-1}}_0 \cdots {M^{-1}}_m D {N^{-1}}_0 \cdots {N^{-1}}_n)^T) = \det(({N^{-1}}_n)^T \cdots ({N^{-1}}_0)^T D^T ({M^{-1}}_m)^T \cdots ({M^{-1}}_0)^T) = \det(D^T) = \det(D) = \det({M^{-1}}_1 \cdots {M^{-1}}_m D {N^{-1}}_1 \cdots {N^{-1}}_n) = \det(A)$.**

2.23 Algorithm 23 (Compound matrix of transpose verification)

Choose an $m \times n$ matrix, A, whose entries are polynomials and an integer $0 \leq k \leq \min(m, n)$. Now do the following:

1. Compute the value $C_k(A^T)$.
2. For each row label I of $C_k(A^T)$, do the following:
   (a) For each column label J of $C_k(A^T)$, do the following:
      i. Let B be a submatrix of A^T formed by selecting the rows I and the columns J.
      ii. Using algorithm 22, verify that $(C_k(A^T))_{I,J} = \det_{I,J}(A^T) = \det(B) = \det(B^T) = \det_{J,I}(A) = (C_k(A))_{J,I}$.
3. **Therefore verify that $(C_k(A))^T = (C_k(A^T))$.**

2.24 Algorithm 24 (Linear system solution construction)

Choose an $m \times n$ matrix, A, and an $m \times p$ matrix, B, both of whose entries are only rationals and do the following:

1. Execute algorithm 6 on A and let $\langle M^{-1}, D, N^{-1}\rangle$ receive the result.
2. Verify that $A = M^{-1}DN^{-1}$.
3. Verify that M^{-1}, D, and N^{-1} contain only rational entries.
4. If the indices of the rows of D that are entirely zero are also the indices of the rows of MB that are entirely zero, then:
   (a) Let C be an $n \times p$ matrix with its i^{th} row given as follows:
      i. If $D_{i,i} \neq 0$, then do the following:
         A. Let row i be row i of MB divided by $D_{i,i}$.
      ii. Otherwise, do the following:
         A. **Choose p rational numbers to fill up the row.**
   (b) Verify that $DC = MB$.
   (c) Let E be NC.
   (d) **Therefore using algorithm 7, verify that $AE = M^{-1}DN^{-1}E = M^{-1}DN^{-1}NC = M^{-1}DI_nC = M^{-1}DC = M^{-1}MB = I_mB = B$.**
   (e) **Yield the tuple $\langle E\rangle$.**

The notation $A\backslash B$ shall be used to refer to the result, E, of invoking algorithm 24 on matrices A and B.

Make an analogous algorithm that yields an F such that $FA = B$. The notation B/A shall be used to refer to the F yielded by invoking this algorithm.

2.25 Algorithm 25

Choose two $m \times m$ matrices, A and B, both of whose entries are only rationals, such that $AB = I_m$ and do the following:

1. Execute algorithm 6 on B and let $\langle M^{-1}, D, N^{-1}\rangle$ receive the result.
2. Verify that $B = M^{-1}DN^{-1}$.
3. If D has a zero on its diagonal, then do the following:
   (a) Using algorithm 19, verify that $\det(I_m) = \det(AB) = \det(A)\det(B) = \det(A)\det(D) = \det(A) * 0 = 0$.
   (b) Using algorithm 8, verify that $\det(I_m) = 1^m = 1$.
   (c) Verify that $0 = 1$.
   (d) **Abort algorithm.**
4. Otherwise do the following:
   (a) Verify that D does not have a zero on its diagonal.
   (b) Verify that $B\backslash I_m = I_m(B\backslash I_m) = AB(B\backslash I_m) = A(B(B\backslash I_m)) = AI_m = A$.
   (c) **Therefore verify that $BA = B(B\backslash I_m) = I_m$.**

2.26 Algorithm 26

Choose $m \times m$ polynomial matrices $\langle B, M, M^{-1}, A, N^{-1}, N\rangle$ such that:

1. **B and A are rational matrices.**
2. **$MM^{-1} = I_m$.**
3. **$N^{-1}N = I_m$.**
4. **$xI_m - B = M(xI_m - A)N$.**

and do the following:

1. Post-multiply both sides of (3) by N^{-1}.
2. Verify that $(xI_m - B)N^{-1} = M(xI_m - A)NN^{-1} = M(xI_m - A)I_m = M(xI_m - A)$.
3. Let $M_0x^b + M_1x^{b-1} + \cdots + M_bx^0 = M$, where the M_i are rational matrices.
4. Now let $R_1 = B^bM_0 + B^{b-1}M_1 + \cdots + B^0M_b$.
5. Verify that $M - R_1 = (xI_m - B)\sum_{k=1}^{b}(x^{k-1}I_mB^0 + x^{k-2}I_mB^1 + \cdots + x^0I_mB^{k-1})M_k$.
6. Let $Q_1 = \sum_{k=1}^{b}(x^{k-1}I_mB^0 + x^{k-2}I_mB^1 + \cdots + x^0I_mB^{k-1})M_k$.
7. Verify that $M = (xI_m - B)Q_1 + R_1$.
8. Let $N^{-1}{}_0x^a + N^{-1}{}_1x^{a-1} + \cdots + N^{-1}{}_ax^0 = N^{-1}$ where the $N^{-1}{}_j$ are rational matrices.
9. Let $R_2 = N^{-1}{}_0A^a + N^{-1}{}_1A^{a-1} + \cdots + N^{-1}{}_aA^0$.

10. Verify that $N^{-1} - R_2 = \sum_{k=1}^{a} N^{-1}{}_k(x^{k-1}I_mA^0 + x^{k-2}I_mA^1 + \cdots + x^0I_mA^{k-1})(xI_m - A)$.
11. Let $Q_2 = \sum_{k=1}^{a} N^{-1}{}_k(x^{k-1}I_mA^0 + x^{k-2}I_mA^1 + \cdots + x^0I_mA^{k-1})$.
12. Verify that $N^{-1} = Q_2(xI_m - A) + R_2$.
13. By substituting M and N^{-1} into (2), verify that $(xI_m - B)(Q_2(xI_m - A) + R_2) = ((xI_m - B)Q_1 + R_1)(xI_m - A)$.
14. By rearranging both sides, verify that $(xI_m - B)(Q_2 - Q_1)(xI_m - A) = R_1(xI_m - A) - (xI_m - B)R_2$.
15. By equating the coefficients of different powers of x both sides, verify that $Q_2 - Q_1 = 0_{m\times m}$.
16. Verify that $R_1(xI_m - A) - (xI_m - B)R_2 = (xI_m - B)(Q_2 - Q_1)(xI_m - A) = (xI_m - B)0_{m\times m}(xI_m - A) = 0_{m\times m}$.
17. Therefore by adding $(xI_m - B)R_2$ to both sides, verify that $xR_1 - R_1A = R_1(xI_m - A) = (xI_m - B)R_2 = xR_2 - BR_2$.
18. By equating the coefficients of x on both sides, verify that $R_1 = R_2$.
19. Therefore verify that $xR_1 - R_1A = R_1(xI_m - A) = (xI_m - B)R_2 = (xI_m - B)R_1 = xR_1 - BR_1$.
20. Therefore by adding $-xR_1$ to both sides, verify that $-R_1A = -BR_1$
21. Therefore by negating both sides, verify that $R_1A = BR_1$.
22. In a similar way to (8), construct a polynomial matrix Q_3, and a rational matrix R_3 such that $M^{-1} = (xI - A)Q_3 + R_3$.
23. Verify that $I_m = MM^{-1} = ((xI_m - B)Q_1 + R_1)M^{-1} = (xI_m - B)Q_1M^{-1} + R_1M^{-1} = (xI_m - B)Q_1M^{-1} + R_1(xI - A)Q_3 + R_1R_3 = (xI_m - B)Q_1M^{-1} + (xI - B)R_1Q_3 + R_1R_3 = (xI_m - B)(Q_1M^{-1} + R_1Q_3) + R_1R_3$.
24. By equating the powers of x on both sides, verify that $Q_1M^{-1} + R_1Q_3 = 0$.
25. By substituting zero for $Q_1M^{-1} + R_1Q_3$, **verify that $I_m = (xI_m - B)0_{m\times m} + R_1R_3 = R_1R_3$.**
26. **Therefore, verify that $B = BI_m = BR_1R_3 = R_1AR_3$.**
27. **Yield the pair (R_1, R_3).**

2.27 Algorithm 27

Choose an $m \times n$ matrix, A, whose entries are polynomials. Choose two integers $1 \leq i, j \leq m$ such that $i \neq j$ and do the following:

1. Subtract row j from row i.
2. Add row i to row j.
3. Subtract row j from row i.
4. **Verify that the matrix obtained is the same as the original A with its i^{th} row negated and swapped with the j^{th} row.**

2.28 Algorithm 28

Choose an $m \times m$ matrix, A, whose entries are only rationals and do the following:

1. Using algorithm 8, verify that $\det(xI - A)$ is a monic polynomial of degree m.
2. Execute algorithm 3 on the polynomial matrix $xI - A$ and let $\langle B\rangle$ be the result.
3. If any of the diagonal entries of B equal zero, then do the following:
   (a) Using algorithm 8, verify that $\det(B) = 0$.
   (b) Therefore using algorithm 19, verify that $\det(xI - A) = 0$.
   (c) **Abort algorithm.**
4. Otherwise do the following:
   (a) Verify that none of the diagonal entries of B equal zero.
   (b) If the last diagonal entry of B is not monic, then do the following:
      i. Cognizant of the execution of algorithm 3, verify that the first $m - 1$ diagonal entries of B are monic.
      ii. Using algorithm 8, verify that $\det(B)$ equals the product of the diagonal entries of B.
      iii. Therefore verify that $\det(B)$ is not monic
      iv. Therefore by algorithm 19, verify that $\det(xI - A)$ is not monic

v. **Abort algorithm.**

(c) Otherwise do the following:

i. Verify that the last diagonal entry of B is monic.

ii. **Verify that none of the diagonal entries of B equal zero.**

iii. **Verify that all the diagonal entries of B are monic.**

2.29 Algorithm 29 (Rational canonical form construction)

Choose an $m \times m$ matrix, A, whose entries are only rationals and do the following:

1. Execute algorithm 5 on the polynomial matrix $xI_m - A$ and let $\langle B, , \rangle$ be the result.
2. Execute algorithm 28 on A.
3. Let E and F be a 0×0 matrices.
4. **Now iterate through the diagonal entries $p = x^k + p_1 x^{k-1} + p_2 x^{k-2} + \cdots + p_k x^0$ of B and:**

   (a) **If $k > 0$:**

   i. Make a $k \times k$ matrix C.

   ii. Let C's first $k-1$ columns be filled with the last $k-1$ columns of I_k.

   iii. Let C's last column from top to bottom be $-p_k, -p_{k-1}, \cdots, -p_1$.

   iv. Add k columns filled with zeros to the right end of E.

   v. Add k rows filled with zeros to the bottom end of E.

   vi. Set the bottom-right corner of E equal to C.

   vii. Let the matrix $D = xI_k - C$ be our working matrix.

   viii. For $i = k$ going down to $i = 2$, add x times row i to row $i - 1$.

   ix. Verify that D's first $k - 1$ columns are now the last $k - 1$ columns of $-I_k$.

   x. Verify that D's last column is p followed by some other polynomials.

   xi. For $i = 2$ going up to $i = k$, subtract $D_{i,k}$ times column $i-1$ from column k.

   xii. Verify that D's last column is now p followed by zeros.

   xiii. For $i = 2$ going up to $i = k$, negate row $i - 1$ and exchange it with row i using algorithm 27.

   xiv. Verify that D is now a diagonal matrix whose first $k - 1$ diagonal entries are 1 and whose last diagonal entry is p.

   xv. Add k columns filled with zeros to the right end of F.

   xvi. Add k rows filled with zeros to the bottom end of F.

   xvii. Set the bottom-right corner of F equal to D.

   (b) Otherwise if $k = 0$, then do the following:

   i. Verify that p is monic.

   ii. Verify that $p = 1$.

   (c) Otherwise do the following:

   i. **Abort algorithm.**

5. Let n be the sum of the positive degrees of the polynomials on the diagonal of B.
6. Verify that E and F are $n \times n$ matrices.
7. Using algorithm 8, verify that $n = \sum_{i=1}^{i=m} \deg(B_{i,i}) = \deg(\det(B)) = \deg(\det(xI - A)) = m$.
8. Therefore verify that E and F are $m \times m$ matrices.
9. **Based on operations (4aviii) to (4axiii), use row and column operations to transform $xI_m - E$ into F.**
10. **Yield the tuple $\langle E, F \rangle$.**

2.30 Algorithm 30

Choose an $m \times m$ matrix, A, whose entries are only rationals and do the following:

1. Execute algorithm 5 on the polynomial matrix $xI_m - A$ and let $\langle B, {M_3}^{-1}, {N_3}^{-1} \rangle$ be the result.
2. Verify that ${M_3}^{-1}(xI_m - A){N_3}^{-1} = B$.

3. Execute algorithm 6 on the polynomial matrix $xI_m - A$ and let $\langle M_3, B, N_3\rangle$ be the result.

4. Verify that $xI_m - A = M_3BN_3$.

5. Using algorithm 7, verify that $M_3{M_3}^{-1} = I_m$.

6. Using algorithm 7, verify that ${N_3}^{-1}N_3 = I_m$.

7. Augment (9) in an analogous way to how algorithm 5 augments algorithm 3. Let $\langle F, M_1, N_1\rangle$ receive the result once (9) has executed.

8. Augment (9) in an analogous way to how algorithm 6 augments algorithm 3. Let $\langle {M_1}^{-1}, F, {N_1}^{-1}\rangle$ receive the result once (9) has executed.

9. Execute algorithm 29 on the matrix A and let $\langle E, F\rangle$ receive the result.

10. Verify that $M_1(xI_m - E)N_1 = F$.

11. Using algorithm 7, verify that $M_1{M_1}^{-1} = I_m$.

12. Using algorithm 7, verify that ${N_1}^{-1}N_1 = I_m$.

13. Augment (18) in an analogous way to how algorithm 5 augments algorithm 3. Let $\langle B, M_2, N_2\rangle$ receive the result once (18) has executed.

14. Augment (18) in an analogous way to how algorithm 6 augments algorithm 3. Let $\langle {M_2}^{-1}, B, {N_2}^{-1}\rangle$ receive the result once (18) has executed.

15. Verify that F and B have the same positive degree polynomials on their diagonals.

16. Verify that the rest of the diagonals of F and B are 1s.

17. Verify that F is the same as B up to rearrangement of the diagonal entries.

18. Rearrange a copy of F to be B by using diagonal entry swaps. In general, swap the i^{th} and j^{th} diagonal entries as follows:

    (a) Use algorithm 27 to negate row i and swap it with row j.

    (b) Use an algorithm analogous to algorithm 27 to negate column i and swap it with column j.

19. Verify that $M_2FN_2 = B$.

20. Using algorithm 7, verify that $M_2{M_2}^{-1} = I_m$.

21. Using algorithm 7, verify that ${N_2}^{-1}N_2 = I_m$.

22. Let $M = M_3M_2M_1$.

23. Let $N = N_1N_2N_3$.

24. Verify that $xI_m - A = M_3BN_3 = M_3M_2FN_2N_3 = M_3M_2M_1(xI_m - E)N_1N_2N_3 = M(xI_m - E)N$.

25. Let $M^{-1} = {M_1}^{-1}{M_2}^{-1}{M_3}^{-1}$.

26. Let $N^{-1} = {N_3}^{-1}{N_2}^{-1}{N_1}^{-1}$.

27. Verify that $MM^{-1} = M_3M_2M_1{M_1}^{-1}{M_2}^{-1}{M_3}^{-1} = I_m$.

28. Verify that $N^{-1}N = {N_3}^{-1}{N_2}^{-1}{N_1}^{-1}N_1N_2N_3 = I_m$.

29. **Execute algorithm 26 on the matrices $\langle A, M, M^{-1}, E, N^{-1}, N\rangle$. Let the tuple $\langle R_1, R_3\rangle$ be the result.**

30. Verify that $A = R_1ER_3$.

31. Verify that $R_1R_3 = I_m$.

32. **Yield the tuple $\langle R_1, E, R_3\rangle$.**

2.31 Algorithm 31 (Block matrix multiplication)

Choose an $m\times n$ matrix, A, and an $n\times k$ matrix, B, whose entries are polynomials. Choose integers $1 \le a \le m$, $1 \le b \le n$, and $1 \le c \le k$. Now do the following:

1. Let C be the submatrix of A that spans rows 1 to $a-1$ and columns 1 to $b-1$.

2. Let D be the submatrix of A that spans rows 1 to $a-1$ and columns b to n.

3. Let E be the submatrix of B that spans rows 1 to $b-1$ and columns 1 to $c-1$.

4. Let F be the submatrix of A that spans rows b to n and columns 1 to $c-1$.

5. Multiply matrix A by matrix B.

6. For each $1 \le i \le a-1$, do the following:

    (a) For each $1 \le j \le c-1$, do the following:

        i. Verify that $(AB)_{i,j} = \sum_{p=1}^{n} A_{i,p}B_{p,j} = \sum_{p=1}^{b-1} A_{i,p}B_{p,j} + \sum_{p=b}^{n} A_{i,p}B_{p,j} = \sum_{p=1}^{b-1} C_{i,p}E_{p,j} + \sum_{p=1}^{1+n-b} D_{i,p}F_{p,j} = (CE)_{i,j} + (DF)_{i,j}$.

7. **Therefore verify that the top left $(a-1) \times (c-1)$ block of AB equals $CE + DF$.**

8. **Do similar computations to verify that the other three blocks of AB are computed in an analogous way to multiplying two 2×2 matrices.**

2.32 Algorithm 32

Choose an $m \times m$ matrix, A, whose entries are only rationals and do the following:

1. Execute algorithm 3 on the polynomial matrix $xI - A$ and let B be the result.

2. Execute algorithm 28 on A.

3. Let $r = x^t + r_1 x^{t-1} + r_2 x^{t-2} + \cdots + r_t x^0 = B_{m,m}$.

4. Execute algorithm 30 on the matrix A to obtain the matrix E.

5. Verify that $R_3 R_1 = I_m$.

6. Using algorithm 30, verify that $r(A) = A^t + r_1 A^{t-1} + r_2 A^{t-2} + \cdots + r_t A^0 = (R_1 E R_3)^t + r_1 (R_1 E R_3)^{t-1} + r_2 (R_1 E R_3)^{t-2} + \cdots + r_t (R_1 E R_3)^0 = R_1 (E^t + r_1 E^{t-1} + r_2 E^{t-2} + \cdots + r_t E^0) R_3 = R_1 r(E) R_3$.

7. For $i = 0$ up to $i = t$, by repeated applications of algorithm 31, verify that E^i evaluates to E with all its diagonal blocks exponentiated to i.

8. Therefore verify that $r(E)$ evaluates to $m \times m$ matrix whose diagonal blocks are the application of r on the corresponding diagonal blocks of E.

9. For $j = 1$ to $j = m$:

   (a) If $\deg(B_{j,j}) > 0$, then do the following:

   (b) Let $p = x^k + p_1 x^{k-1} + p_2 x^{k-2} + \cdots + p_k x^0 = B_{j,j}$.

   (c) Let G be the corresponding $k \times k$ block on the diagonal of E.

   (d) Let e_i denote a $k \times 1$ matrix that is 0, except for its i^{th} entry which is 1.

   (e) Then by G's construction, for $i = 1$ up to $i = k$, verify that $G^{i-1} e_1 = G^{i-2} e_2 = \cdots = G^0 e_i = e_i$.

   (f) Let $0_{m \times n}$ denote an $m \times n$ matrix of zeros.

   (g) Therefore, for $i = 1$ up to $i = k$: Cognizant of the construction of G's last column, verify that $p(G) e_i = (G^k + p_1 G^{k-1} + p_2 G^{k-2} + \cdots + p_k G^0) e_i = (G^k + p_1 G^{k-1} + p_2 G^{k-2} + \cdots + p_k G^0) G^{i-1} e_1 = G^{i-1} (G G^{k-1} + p_1 G^{k-1} + p_2 G^{k-2} + \cdots + p_k G^0) e_1 = G^{i-1} (G e_k + p_1 e_k + p_2 e_{k-1} + \cdots + p_k e_1) = G^{i-1} 0_{k \times 1} = 0_{k \times 1}$.

   (h) Therefore verify that $p(G) = 0_{k \times k}$.

   (i) Using the execution of algorithm 3 in (1), verify that $r = B_{m,m} = B_{j,j} u_{j+1} u_{j+2} \cdots u_m = B_{j,j} q = pq$, where $q = u_{j+1} u_{j+2} \cdots u_m$.

   (j) Therefore verify that $r(G) = p(G) q(G) = 0_{k \times k} q(G) = 0_{k \times k}$.

10. Therefore verify that the diagonal blocks of $r(E)$ each equal $0_{i \times i}$, where i is the size of each diagonal block.

11. Therefore verify that $r(E) = 0_{m \times m}$.

12. **Therefore verify that $r(A) = R_1 r(E) R_3 = R_1 0_{m \times m} R_3 = 0_{m \times m}$.**

2.33 Algorithm 33

Choose an $m \times m$ matrix, A, whose entries are only rationals. Choose a non-zero polynomial $p = x^t + p_1 x^{t-1} + p_2 x^{t-2} + \cdots + p_t x^0$ such that $p(A) = 0$.

1. Execute algorithm 3 on the polynomial matrix $xI - A$ and let B be the result.

2. Execute algorithm 28 on A.

3. Let $r = x^u + r_1 x^{u-1} + r_2 x^{u-2} + \cdots + r_u x^0 = B_{m,m}$.

4. Execute algorithm 30 on the matrix A to obtain the matrix E.

5. Let F be a 1×2 matrix consisting in-order of p and r.

6. Execute algorithm 5 on F and let $\langle D, M, N \rangle$ receive the result.

7. Execute algorithm 6 on F and let $\langle M^{-1}, D, N^{-1} \rangle$ receive the result.

8. Let $g = x^w + g_1 x^{w-1} + g_2 x^{w-2} + \cdots + g_w x^0 = D_{1,1}$.

9. If $\deg(p) \neq \deg(r)$, do the following:

(a) Verify that the execution of the inner algorithm 1 begun with repeated executions of (2c) to bring the degree of the higher degree polynomial down to that of the lower.

(b) Verify that afterwords, each iteration of (2) lowered the degree of either entry.

10. Therefore verify that $\deg(g) \leq \min(\deg(p), \deg(r))$.

11. Therefore verify that $w \leq u$.

12. Verify that $D = MFN$.

13. Therefore verify that $g = D_{1,1} = N_{1,1}p + N_{2,1}r$.

14. Therefore using algorithm 32, verify that $g(A) = N_{1,1}(A)p(A) + N_{2,1}(A)r(A) = N_{1,1}(A)0_{m\times m} + N_{2,1}(A)0_{m\times m} = 0_{m\times m}$.

15. Using steps of algorithm 32, algorithm 25, and algorithm 30, verify that $g(E) = I_m g(E) I_m = R_3R_1g(E)R_3R_1 = R_3g(A)R_1 = R_30_{m\times m}R_1 = 0_{m\times m}$.

16. Let G be the $u \times u$ block in E corresponding to the polynomial r in B.

17. Using steps of algorithm 32, verify that $g(G) = 0_{u\times u}$.

18. Therefore verify, using steps of algorithm 32, that $g(G)e_1 = (G^w + g_1G^{w-1} + g_2G^{w-2} + \cdots + g_wG^0)e_1 = Ge_w + g_1e_w + g_2e_{w-1} + \cdots + g_we_1$.

19. If $w < u$, then:

    (a) Verify that $Ge_w + g_1e_w + g_2e_{w-1} + \cdots + g_we_1 = e_{w+1} + g_1e_w + g_2e_{w-1} + \cdots + g_we_1 = 0_{u\times 1}$.

    (b) Therefore verify that $1 = 0$.

    (c) **Abort algorithm.**

20. Otherwise it should be that $w = u$. Now:

    (a) Verify that $Ge_w + g_1e_w + g_2e_{w-1} + \cdots + g_we_1 = Ge_u + g_1e_u + g_2e_{u-1} + \cdots + g_ue_1 = 0_{u\times 1}$.

    (b) Therefore for $i = 1$ to $i = u$, do the following:

        i. Verify that $-r_i + g_i = 0$.

        ii. Therefore verify that $g_i = r_i$.

    (c) **Therefore verify that $g = r$.**

    (d) Verify that $F = M^{-1}DN^{-1}$.

    (e) **Therefore verify that $p = F_{1,1} = D_{1,1}N^{-1}{}_{1,1} + D_{1,2}N^{-1}{}_{2,1} = N^{-1}{}_{1,1}g + N^{-1}{}_{2,1} * 0 = N^{-1}{}_{1,1}g = N^{-1}{}_{1,1}r$.**

2.34 Algorithm 34 (Difference of powers)

Choose an integer $n > 0$ and a formal polynomial $p = p_0x^n + p_1x^{n-1} + \cdots + p_n$.

1. Let the formal polynomial $G(y,z) = \sum_{r=1}^{n} p_{n-r}(z^{r-1} + z^{r-2}y + \cdots + zy^{r-2} + y^{r-1})$.
2. **Verify that the formal polynomial $p(z) - p(y) = (p_0z^n + p_1z^{n-1} + \cdots + p_n) - (p_0y^n + p_1y^{n-1} + \cdots + p_n) = (\sum_{r=0}^{n} p_{n-r}z^r) - (\sum_{r=0}^{n} p_{n-r}y^r) = \sum_{r=1}^{n} p_{n-r}(z^r - y^r) = \sum_{r=1}^{n} p_{n-r}(z-y)(z^{r-1} + z^{r-2}y + \cdots + zy^{r-2} + y^{r-1}) = (z-y)\sum_{r=1}^{n} p_{n-r}(z^{r-1} + z^{r-2}y + \cdots + zy^{r-2} + y^{r-1}) = (z-y)G(y,z)$.**
3. **Yield the tuple $\langle G(y,z)\rangle$.**

2.35 Algorithm 35

Choose a formal polynomial $p = x^n + p_1x^{n-1} + \cdots + p_n$ and rationals $a_1 < a_2 < \cdots < a_n < a_{n+1}$ in such a way that for $i = 1$ to $i = n+1$, $p(a_i) = 0$. Now do the following:

1. Write p as $1 * p$, so that it has two factors.
2. For $i = 1$ up to $i = n$, do the following:

    (a) Let g be the rightmost factor of p.

    (b) If $g(a_i) \neq 0$, do the following:

        i. For $k = 1$ to $k = i-1$, verify that $(a_i - a_k) \neq 0$.

        ii. Verify that $p(a_i) \neq 0$.

        iii. **Abort algorithm.**

    (c) Otherwise $g(a_i) = 0$. Now do the following:

        i. Execute algorithm 34 on g and let $(G(x,y))$ be the result.

        ii. Let the formal polynomial $q = q(x) = G(a_i, x)$.

        iii. Verify that the formal polynomial $g = g(x) = g(x) - g(a_i) = (x-a_i)G(a_i,x) = (x-a_i)q(x) = (x-a_i)q$.

iv. Verify that $p = (x-a_1)(x-a_2)\cdots(x-a_i)q$.

3. Now verify that $p = (x-a_1)(x-a_2)\cdots(x-a_n)1$.
4. Using (3), verify that $p(a_{n+1}) \neq 0$.
5. **Abort algorithm.**

2.36 Algorithm 36 (Bisection)

Choose a formal polynomial f. Choose rational numbers $a < b$ such that $\operatorname{sgn}(f(a)) = -\operatorname{sgn}(f(b))$. Choose a rational number target $B > 0$. Now do the following:

1. Execute algorithm 34 on f and let $(G(x,y))$ be the result.
2. Verify that the formal polynomial $f(y) - f(x) = (y-x)G(x,y)$.
3. Let G' be G but with all negative signs replaced with positive signs.
4. Let $U = \max(G'(|a|,|a|), G'(|b|,|b|))$.
5. Until $|b-a|U < B$
   (a) Let $c = \frac{a+b}{2}$.
   (b) If $\operatorname{sgn}(f(a)) = -\operatorname{sgn}(f(c))$, then:
      i. Let $b = c$.
   (c) Otherwise if $\operatorname{sgn}(f(c)) = -\operatorname{sgn}(f(b))$, then:
      i. Let $a = c$.
   (d) Otherwise if $f(c) = 0$, then do the following:
      i. Record the rational number c.
      ii. If less than or equal to i rational numbers have been recorded, then:
         A. Let $a = c$.
         B. Go to (1bi).
      iii. Otherwise the $i+1$ numbers c_k such that $c_1 < c_2 < \cdots < c_i < c_{i+1}$ have been recorded. Now do the following:
         A. Execute algorithm 35 on the formal polynomial f_i and the rationals $c_1 < c_2 < \cdots < c_i < c_{i+1}$.
         B. **Abort algorithm.**
   (e) Otherwise, do the following:
      i. Verify that $f(a) \not< 0$.
      ii. Verify that $f(a) \neq 0$.
      iii. Verify that $f(a) \not> 0$.
      iv. **Abort algorithm.**
6. **Verify that $|f(a)|, |f(b)| < |f(b) - f(a)| = |(b-a)G(a,b)| < |(b-a)G'(a,b)| < |b-a|U < B$.**

2.37 Algorithm 37

Choose a polynomial $f_n = x^n + p_1x^{n-1} + \cdots + p_n$ and pairs of rationals $(a_n, b_n), (a_{n-1}, b_{n-1}), \cdots, (a_0, b_0)$ in such a way that:

1. $a_n < b_n \leq a_{n-1} < b_{n-1} \leq \cdots \leq a_1 < b_1 \leq a_0 < b_0$**.**

2. For $i = 0$ **to** $i = n$**,** $\operatorname{sgn}(f_n(a_i)) = -\operatorname{sgn}(f_n(b_i))$**.**

Now do the following:

1. For $i = n$ to $i = 1$:
   (a) Let $B = \min_{k=0}^{i-1} \min(|f_i(a_i)|, |f_i(b_i)|)$.
   (b) Execute algorithm 36 on the formal polynomial f_i, interval (a_i, b_i), and target of B. Let a_i and b_i receive their updates.
   (c) Verify that $|f_i(b_i)| < B$.
   (d) f should evaluate to $f - f_i(b_i) + f_i(b_i)$ which should evaluate to $(x-b_i)f_{i-1} + f_i(b_i)$, where f_{i-1} is a monic $(i-1)^{th}$ degree formal polynomial.
   (e) For $k = 0$ to $k = i-1$, do the following:
      i. If $f_i(a_k) > B$, verify that:
         A. $f_i(a_k) > B > |f_i(b_i)| \geq f_i(b_i)$.
         B. $f_i(a_k) - f_i(b_i) > 0$.
         C. $(a_k - b_i)f_{i-1}(a_k) > 0$.
         D. $f_{i-1}(a_k) > 0$.
         E. $f_i(b_k) < -B < -|f_i(b_i)| \leq f_i(b_i)$.
         F. $f_i(b_k) - f_i(b_i) < 0$.
         G. $(b_k - b_i)f_{i-1}(b_k) < 0$.
         H. $f_{i-1}(b_k) < 0$.
      ii. Otherwise, if $f_i(a_k) < -B$, verify that:
         A. $f_{i-1}(a_k) < 0$.

B. $f_{i-1}(b_k) > 0$.

iii. Otherwise:

A. **Verify that 0=1.**

2. By (1d), f_0 should equal 1.

3. By (1eiD) and (1eiH) or (1eiiA) and (1eiiB), $\operatorname{sgn}(f_0(a_0)) = -\operatorname{sgn}(f_0(b_0))$.

4. **Verify that 0=1.**

2.38 Algorithm 38 (Sturm's algorithm initialization)

Choose a sequence of polynomials $s_0 = s_{00}, s_1 = s_{10}x^1 + s_{11}, \cdots, s_m = s_{m0}x^m + s_{m1}x^{m-1} + \cdots + s_{mm}$ **and another sequence of polynomials** $q_1, q_2, \cdots, q_{m-1}$ **in such a way that**

1. **For** $i = 0$ **to** $i = m$**,** $s_{i0} > 0$

2. **For** $i = 1$ **to** $i = m-1$**,** $s_{i-1} + s_{i+1} = q_i s_i$**.**

Now do the following:

1. Let $J_i(x)$ be a shorthand for the number of sign changes in the sequence $s_0(x), s_1(x), \cdots, s_i(x)$.

2. For $i = 1$ to $i = m$, do the following:

   (a) Execute algorithm 34 on s_i and let $(G_i(x, y))$ be the result.

   (b) **Verify that the formal polynomial** $s_i(y) - s_i(x) = (y - x)G_i(x, y)$**.**

3. For $i = 1$ to $i = m - 1$, do the following:

   (a) It should be that $q_i s_i - s_{i+1} = s_{i-1}$. Make this our working equation.

   (b) If $i > 1$:

      i. Substitute this s_{i-1} into the equation mentioned in (7c) of the previous iteration. Let the outcome be our working equation.

   (c) Our working equation should now be in terms of s_i, s_{i+1}, and s_0.

   (d) **Factorize this equation and divide it through by** s_0 **to obtain the equation** $g_i s_{i+1} + h_i s_i = 1$**, where** g_i **and** h_i **are polynomials.**

2.39 Algorithm 39 (Change in number of sign changes verification)

1. **Execute algorithm 38.**

2. **Choose rational numbers** c **and** d **in such a way that:**

   (a) $J_m(c)$ **and** $J_m(d)$ **are well defined.**

   (b) Letting $B = \max_{i=1}^{m} |G_i(c, d)|$**.**

   (c) Letting $C = \max_{i=1}^{m-1} \max(|g_i(c)|, |h_i(c)|, |g_i(d)|, |h_i(d)|)$**.**

   (d) Letting $D = \max_{i=1}^{m-1} \max(|q_i(c)|, |q_i(d)|)$**.**

   (e) $|d - c| < \frac{1}{BCD}$**.**

3. Let $i = 0$.

4. Do the following:

   (a) $\operatorname{sgn}(s_i(c))$ should equal $\operatorname{sgn}(s_i(d))$.

   (b) $J_i(c)$ should equal $J_i(d)$.

   (c) If $\operatorname{sgn}(s_{i+1}(c)) = \operatorname{sgn}(s_{i+1}(d))$, verify the following:

      i. $J_{i+1}(c) = J_{i+1}(d)$.

      ii. Set i to $i + 1$ and go to (4) if the new $i < m$.

   (d) Otherwise, if $\operatorname{sgn}(s_{i+1}(c)) \neq \operatorname{sgn}(s_{i+1}(d))$ and $i + 2 \leq m$, verify the following:

      i. $|s_{i+1}(c)| < |s_{i+1}(c) - s_{i+1}(d)| = |c - d||G_{i+1}(c, d)| < |c - d|B < lB = \frac{1}{CD}$

      ii. $|s_{i+1}(d)| < \frac{1}{CD}$

      iii. $1 = g_{i+1}(c)s_{i+2}(c) + h_{i+1}(c)s_{i+1}(c) = |g_{i+1}(c)s_{i+2}(c) + h_{i+1}(c)s_{i+1}(c)| \leq |g_{i+1}(c)||s_{i+2}(c)| + |h_{i+1}(c)||s_{i+1}(c)| < C(|s_{i+2}(c)| + \frac{1}{CD})$

      iv. $\frac{1}{C}(1 - \frac{1}{D}) < |s_{i+2}(c)|$

      v. $\frac{1}{C}(1 - \frac{1}{D}) < |s_{i+2}(d)|$

      vi. If $\operatorname{sgn}(s_{i+2}(c)) \neq \operatorname{sgn}(s_{i+2}(d))$, verify the following:

         A. $|s_{i+2}(c)| < \frac{1}{CD} = \frac{1}{C} \cdot \frac{1}{D} < \frac{1}{C}(1 - \frac{1}{D})$

         B. **Verify that 0=1.**

      vii. Otherwise if $\operatorname{sgn}(s_i(c)) = \operatorname{sgn}(s_{i+2}(c))$, verify the following:

A. $2\frac{1}{C}(1 - \frac{1}{D}) \leq |s_i(c)| + |s_{i+2}(c)| = |s_i(c) + s_{i+2}(c)| = |q_{i+1}(c)s_{i+1}(c)| < D\frac{1}{CD}$.

B. $2(1 - \frac{1}{D}) < 1$.

C. $D < 2$.

D. **Verify that 0=1.**

viii. Otherwise $\text{sgn}(s_{i+2}(c)) = \text{sgn}(s_{i+2}(d))$ and $\text{sgn}(s_i(c)) \neq \text{sgn}(s_{i+2}(c))$. Now verify the following:

A. $J_{i+2}(c) = J_{i+2}(d)$.

B. Set i to $i+2$ and go to (4).

(e) Otherwise $\text{sgn}(s_{i+1}(c)) \neq \text{sgn}(s_{i+1}(d))$ and $i+1 = m$. Now verify the following:

i. $|s_{i+1}(c)| < \frac{1}{CD}$.

ii. $|s_{i+1}(d)| < \frac{1}{CD}$.

iii. $|J_{i+1}(c) - J_{i+1}(d)| = 1$.

5. **If** $\text{sgn}(s_m(c)) = \text{sgn}(s_m(d))$**, then** $J_m(c)$ **should equal** $J_m(d)$**. Otherwise** $|J_m(d) - J_m(c)|$ **should equal** 1**.**

2.40 Algorithm 40 (Cauchy's positive verification)

Choose a non-zero polynomial $p = p_0x^t + p_1x^{t-1} + p_2x^{t-2} + \cdots + p_tx^0$**, where** $p_0 > 0$**. Choose a rational** $k > 1 + \max_{i=1}^t |\frac{p_i}{p_0}|$**.** In reverse order verify the following:

1. $p(k) > 0$
2. $p_0k^n + p_1k^{n-1} + \cdots + p_nk^0 > 0$
3. $k^n + \frac{p_1}{p_0}k^{n-1} + \cdots + \frac{p_n}{p_0}k^0 > 0$
4. $k^n > -(\frac{p_1}{p_0}k^{n-1} + \cdots + \frac{p_n}{p_0}k^0)$
5. $k^n > |\frac{p_1}{p_0}k^{n-1} + \cdots + \frac{p_n}{p_0}k^0|$
6. $k^n > |\max_{i=1}^t |\frac{p_i}{p_0}|(k^{n-1} + \cdots + k^0)|$
7. $k^n > \max_{i=1}^t |\frac{p_i}{p_0}|\frac{k^n - 1}{k-1}$
8. $k^{n+1} - k^n > \max_{i=1}^t |\frac{p_i}{p_0}|(k^n - 1)$
9. $k^{n+1} - (1 + \max_{i=1}^t |\frac{p_i}{p_0}|)k^n + \max_{i=1}^t |\frac{p_i}{p_0}| > 0$
10. $k > 1 + \max_{i=1}^t |\frac{p_i}{p_0}|$

2.41 Algorithm 41 (Cauchy's alternation verification)

Choose a non-zero polynomial $p = p_0x^t + p_1x^{t-1} + p_2x^{t-2} + \cdots + p_tx^0$**, where** $p_0 > 0$**. Choose a rational** $k < -(1 + \max_{i=1}^t |\frac{p_i}{p_0}|)$**.** Now do the following:

1. Let $q = q_0x^t + q_1x^{t-1} + q_2x^{t-2} + \cdots + q_tx^0$, where $q_i = (-1)^i p_i$.
2. It should be that $k < -(1 + \max_{i=1}^t |\frac{q_i}{q_0}|)$.
3. Therefore it should be that $-k > 1 + \max_{i=1}^t |\frac{q_i}{q_0}|$.
4. Now execute algorithm 38 on q and $-k$.
5. It should be that $q(-k) > 0$, that is, $\sum_{i=0}^t q_i(-k)^{t-i} > 0$.
6. Therefore, it should be that $\sum_{i=0}^t (-1)^i(-1)^{t-i}p_ik^{t-i} > 0$.
7. Therefore, it should be that $(-1)^t \sum_{i=0}^t p_ik^{t-i} > 0$.
8. **Therefore it should be that** $(-1)^tp(k) > 0$**.**

2.42 Algorithm 42 (Sturm's sign change)

1. **Execute algorithm 38.**
2. Let $U = 1 + \max_{i=0}^m \left(1 + \max_{j=1}^i |\frac{s_{ij}}{s_{i0}}|\right)$
3. By algorithm 40, $J(U)$ should evaluate to 0.
4. By algorithm 41, $J(-U)$ should evaluate to m.
5. For $i = 1$ to $i = m$: Let $G'_i(x, y)$ be $G_i(x, y)$ with all negative signs replaced with positive signs.
6. Let the rational $B = \max_{i=1}^m G'_i(U, U)$.
7. For $i = 1$ to $i = m-1$: Let the polynomial g'_i be g_i with all negative signs replaced with positive signs. Let the polynomial h'_i be h_i with all negative signs replaced with positive signs.
8. Let $C = C = \max_{i=1}^m \max(g'_i(U), h'_i(U))$.
9. For $i = 1$ to $i = m-1$: Let the polynomial q'_i be q_i with all negative signs replaced with positive signs.
10. Let $D = \max(3, \max_{i=1}^m q'_i(U))$
11. Let $l = \frac{1}{BCD}$ and $c = -U$.

12. Do the following:
    (a) If $c + l \leq U$, then:
        i. Choose a number d in a way such that $c < d < c + l$ and $J(d)$ is well-defined.
    (b) Otherwise let $d = U$.
    (c) Execute algorithm 39.
    (d) **If $J_m(c) \neq J_m(d)$, then record the pair of rational numbers (c, d).**
    (e) Now let $c = d$ and go to (12) if the new $c \neq U$.
13. If less than m pairs of rational numbers were recorded, then verify the following:
    (a) Each change of $J_m(x)$ over the course of (12) was by 1.
    (b) $J_m(x)$ changed less than m times over the course of (12).
    (c) Therefore it should be that $|J_m(U) - J_m(-U)| < m$.
    (d) **Verify that 0=1.**
14. If more than m pairs of rational numbers were recorded, then verify the following:
    (a) **Execute algorithm 37 on the formal polynomial $s_m(x)$ along with the first $m + 1$ recorded pairs.**
15. **Otherwise verify that:**
    (a) Exactly m pairs of rational numbers $(c_1, d_1), (c_2, d_2), \cdots, (c_m, d_m)$ were recorded.
    (b) $c_1 < d_1 \leq c_2 < d_2 \leq \cdots \leq c_m < d_m$.
    (c) For $i = 1$ to $i = m$, $\operatorname{sgn}(s_m(c_i)) = -\operatorname{sgn}(s_m(d_i))$.

2.43 Algorithm 43

Choose an $m \times m$ matrix, A, whose entries are only rationals. Execute algorithm 3 on the polynomial matrix $xI - A$ and let B be the result. Choose a polynomial p such that $\deg(p) < \deg(B_{m,m})$. Now do the following:

1. Execute algorithm 33 on matrix A and polynomial p.
2. Since the inner execution of algorithm 1 at algorithm 33's (5) was decreasing the degree of an entry on every iteration, verify for algorithm 33's g that $\deg(g) \leq \deg(B_{m,m}) - 1$.
3. Since $B_{m,m} = g$, verify that $\deg(B_{m,m}) \leq \deg(B_{m,m}) - 1$.
4. **Abort algorithm.**

2.44 Algorithm 44

Choose an $m \times m$ matrix, A, whose entries are only rationals and do the following:

1. Execute algorithm 3 on the polynomial matrix $xI - A$ and let B be the result.
2. Execute algorithm 28 on A.
3. Let $r = x^t + r_1x^{t-1} + r_2x^{t-2} + \cdots + r_tx^0$ be the last diagonal entry of B.
4. Make an $m^2 * t$ matrix, F, whose i^{th} column is the sequential concatenation of the columns of A^{t-i}.
5. Execute algorithm 5 on F to obtain matrices M, N, and D such that $MFN = D$.
6. Execute algorithm 6 on F to obtain matrices M, N, and D such that $F = M^{-1}DN^{-1}$.
7. Using algorithm 7, it should be that $M^{-1}MFN = I_{m^2}FN = FN = M^{-1}D$.
8. If any column i of N, Ne_i, is equal to zero, then:
    (a) Column i of $N^{-1}N$ should equal zero.
    (b) **Verify that 0=1.**
9. Otherwise if the first entry of $C_t(D)$ is zero, then:
    (a) Some diagonal entry, $D_{i,i}$, of D should be zero.
    (b) Column i of D should be zero.
    (c) Column i of $M^{-1}D$ should be zero.
    (d) Column i of FN should be zero.
    (e) $F(Ne_i)$ should equal zero.
    (f) Therefore $N_{1,i}A^{t-1} + N_{2,i}A^{t-2} + \cdots + N_{t,i}A^0$ should equal zero.
    (g) Execute algorithm 43 on matrix A and polynomial $p = N_{1,i}x^{t-1} + N_{2,i}x^{t-2} + \cdots + N_{t,i}x^0$.

(h) **Verify that 0=1.**

10. Otherwise, if any column i of $C_t(M^{-1})$ is equal to zero, then:

    (a) Column i of $C_t(M)C_t(M^{-1})$ should equal zero.

    (b) $C_t(M)C_t(M^{-1})$ should, by algorithm 18, equal $C_t(MM^{-1})$ which should, by algorithm 7, equal $C_t(I_{m^2})$ which, by algorithm 14, should equal $I_{\binom{m^2}{t}}$.

    (c) Therefore column i of $I_{\binom{m^2}{t}}$ should equal zero.

    (d) **Verify that 0=1.**

11. Otherwise, if column i of $C_t(N^{-1})$ is equal to zero, then:

    (a) **Verify that 0=1.**

12. Otherwise all columns of $C_t(M^{-1})$ are non-zero, the first entry of $C_t(D)$ is non-zero, and $C_t(N^{-1})$ is non-zero. Now verify the following in order:

    (a) $C_t(F) = C_t(M^{-1}DN^{-1})$

    (b) $C_t(F) = C_t(M^{-1})C_t(D)C_t(N^{-1})$

    (c) $C_t(F) = C_t(M^{-1})ke_1C_t(N^{-1})$, where k is a non-zero rational number

    (d) $C_t(F) = kC_t(N^{-1})C_t(M^{-1})e_1$

    (e) $C_t(F) \neq 0_{\binom{m^2}{t}\times 1}$

2.45 Algorithm 45

Choose an $m \times m$ matrix, A, whose entries are only rationals and is such that $A^T = A$. Now do the following:

1. **Execute algorithm 44 on the matrix A.**

2. $C_t(F^TF)$ should evaluate to $C_t(F^T)C_t(F)$ which should evaluate to the sum of squares of the entries of $C_t(F)$ which should be more than zero.

3. Using algorithm 6, it should be possible to evaluate F^TF as $M^{-1}DN^{-1}$.

4. If D has a zero on its diagonal, then:

    (a) $C_t(F^TF)$ should evaluate to $C_t(M^{-1}DN^{-1})$ which should evaluate to $C_t(D)$ which should evaluate to 0.

    (b) **Verify that 0=1.**

5. Otherwise D should not have a zero on its diagonal. So do the following:

    (a) Let $\mathrm{tr}(X)$ be a shorthand for the sum of the diagonal entries of the square matrix X.

    (b) Let $\mathrm{mat}(p_1x^{t-1} + p_1x^{t-2} + \cdots + p_t)$ be a shorthand for $p_1e_1 + p_2e_2 + \cdots + p_te_t$.

    (c) Let $\mathrm{pol}(p_1e_1 + p_2e_2 + \cdots + p_te_t)$ be a shorthand for $p_1x^{t-1} + p_1x^{t-2} + \cdots + p_t$.

    (d) Let $H = (F^TF)\backslash e_1$.

    (e) Separately apply algorithm 5 and algorithm 6 on a 1×2 matrix comprising r followed by $\mathrm{pol}(H)$.

    (f) If d is a monic polynomial of degree is more than 0, then do the following:

        i. From the matrices constructed by algorithm 6, let $d = D_{1,1}$, $b = {N^{-1}}_{1,1}$ and $c = {N^{-1}}_{1,2}$.

        ii. It should be that $r = bd$, where b is a monic polynomial with degree $z < t$.

        iii. It should be that $\mathrm{pol}(H) = cd$, where c is some polynomial.

        iv. Let $u = x^{t-z-1}b$.

        v. $\mathrm{tr}(u(A)h(A))$ should evaluate to $\mathrm{mat}(u)^TF^TFH$ which should evaluate to $\mathrm{mat}(u)^TF^TF((F^TF)\backslash e_1)$ which should evaluate to $\mathrm{mat}(u)^Te_1$ which should evaluate to $\mathrm{mat}(u)_{1,1}$ which should evaluate to 1.

        vi. $\mathrm{tr}(u(A)h(A))$ should also evaluate to $\mathrm{tr}(A^{z-1}b(A)c(A)d(A))$ which should evaluate to $\mathrm{tr}(A^{z-1}c(A)b(A)d(A))$ which should evaluate to $\mathrm{tr}(A^{z-1}c(A)f(A))$ which should evaluate to $\mathrm{tr}(A^{z-1}c(A)0_{m\times m})$ which should evaluate to $\mathrm{tr}(0_{m\times m})$ which should evaluate to 0.

        vii. **Verify that 0=1.**

    (g) Otherwise if $d = 1$, then do the following:

        i. From the matrices constructed by algorithm 6, let $d = D_{1,1}$, $u = N_{1,1}$ and $s_{t+1} = N_{2,1}$.

        ii. **It should be that $uf + s_{t+1}h = 1$.**

(h) Otherwise:

i. **Verify that 0=1.**

2.46 Algorithm 46 (Euclidean division)

Choose two polynomials in x, a and b and do the following:

1. If the degree of a is more than or equal to the degree of b:
   (a) Let y be $\frac{a\text{'s leading coefficient}}{b\text{'s leading coefficient}} x^{a\text{'s degree - }b\text{'s degree}}$
   (b) Let e be $a - yb$. The degree of e should be less than that of the degree of a.
   (c) Apply algorithm 46 on the ordered pair of polynomials e and b. Let c and d be the ordered pair of polynomials yielded by this application.
   (d) It should be that $cb + d = e$ and that d has degree less than b.
   (e) Therefore it should be that $cb + d = a - yb$
   (f) **Therefore it should be that** $(y + c)b + d = a$ **and that** d **has degree less than** b**.**
   (g) **Now yield the ordered pair of polynomials consisting of** $y + c$ **and** d**.**
2. Otherwise:
   (a) **It should be that** $0b + a = a$ **and that** a **has degree less than** b**.**
   (b) **Yield the ordered pair consisting of** 0 **then** a**.**

2.47 Algorithm 47 (Edwards' Sturm chain construction)

Choose an $m \times m$ matrix, A, whose entries are only rationals and is such that $A^T = A$. Now do the following:

1. **Execute algorithm 45 on the matrix A.**
2. If polynomials u and s_{t+1} such that $us_t + s_{t+1}h = 1$ were successfully created, then:
3. Execute algorithm 46 on the ordered pair s_{t+1} and s_t. Let q_t and s_{t-1} be the polynomials yielded by this execution.
4. Verify that $s_{t+1} = q_t s_t + s_{t-1}$, where $\deg(s_{t-1}) < t$.
5. Therefore verify that $us_t + (q_t s_t + s_{t-1})h = 1$.
6. Therefore verify that $u(A)s_t(A) + (q_t(A)s_t(A) + s_{t-1}(A))h(A) = I_{m,m}$.
7. Therefore verify that $s_{t-1}(A)h(A) = I_{m,m}$.
8. Therefore verify that $s_{t-1_0} = \mathrm{tr}(s_{t-1}(A)h(A)) = \mathrm{tr}(I_{m,m}) = m > 0$.
9. For $i = t - 1$ down to $i = 1$, do the following:
   (a) If $i < t - 1$, then do the following:
      i. Verify that $s_i = p_1 s_{t-1} + j_1 s_t$ where $\deg(s_i) = i$, $s_{i0} > 0$, $\deg(p_1) = t-1-i$, and $\deg(j_1) = t - 2 - i$.
      ii. Verify that $s_{i+1} = p_2 s_{t-1} + j_2 s_t$ where $\deg(s_{i+1}) = i$, $\deg(p_2) = t - 2 - i$, and $\deg(j_2) = t - 3 - i$.
   (b) Execute algorithm 46 on the ordered pair $-s_{i+1}$ and $-s_i$. Let q_i and s_{i-1} be the polynomials yielded by this execution.
   (c) Verify that $\deg(q_i) = 1$ and that $q_{i0} = \frac{s_{i+1_0}}{s_{i0}}$.
   (d) Also verify that $-s_{i+1} = -q_i s_i + s_{i-1}$.
   (e) Therefore verify that $q_i s_i = s_{i+1} + s_{i-1}$.
   (f) Therefore verify that $q_i s_i - s_{i+1} = s_{i-1}$.
   (g) If $i < t - 1$, then do the following:
      i. **It should be that** $s_{i-1} = q_i(p_1 s_{t-1} + j_1 s_t) - (p_2 s_{t-1} + j_2 s_t) = p_3 s_{t-1} + j_3 s_t$, **where** $p_3 = q_i p_1 - p_2$ **is of degree** $t - i$ **and** $j_3 = q_i j_1 - j_2$ **is of degree** $t - 1 - i$**.**
   (h) Otherwise:
      i. **Verify that** $s_{i-1} = s_{t-2} = q_{t-1}s_{t-1} - s_t = p_3 s_{t-1} + j_3 s_t$, **where** $p_3 = q_{t-1}$ **of degree** $1 = t - 1$ **and** $j_3 = -1$ **is of degree** $0 = t - 1 - i$**.**
   (i) Therefore verify that $s_{i-1}(A) = p_3(A)s_{t-1}(A) + j_3(A)s_t(A) = p_3(A)s_{t-1}(A) + j_3(A)0_{m\times m} = p_3(A)s_{t-1}(A)$.

(j) If $p_3(A) = 0$, then do the following:

   i. Execute algorithm 43 on the matrix A and polynomial p_3.

   ii. **Abort algorithm.**

(k) Otherwise, if $s_{i-1}(A) = 0_{m\times m}$, then do the following:

   i. Verify that $p_3(A)s_{t-1}(A)h(A) = s_{i-1}(A)h(A) = 0_{m\times m}h(A) = 0_{m\times m}$.

   ii. Verify that $p_3(A)s_{t-1}(A)h(A) = p_3(A)I_{m,m} = p_3(A) \neq 0_{m\times m}$.

   iii. **Abort algorithm.**

(l) Otherwise if $s_{i-1}(A)h(A) = 0_{m\times m}$, then do the following:

   i. Verify that $s_{i-1}(A)h(A)s_{t-1}(A) = 0_{m\times m}s_{t-1}(A) = 0_{m\times m}$.

   ii. Verify that $s_{i-1}(A)h(A)s_{t-1}(A) = s_{i-1}(A)I_{m,m} = s_{i-1}(A) \neq 0$.

   iii. **Abort algorithm.**

(m) Otherwise, do the following:

   i. Verify that $\deg(s_{i-1}) < i$.

   ii. Execute the auxilliary algorithm on the pair (s_{i-1}, s_{i-1}).

   iii. Now verify that $\mathrm{tr}(s_{i-1}(A)^2h(A)^2) = \frac{s_{i-1_0}}{s_{i0}}$.

   iv. Verify that $s_{i-1}(A)h(A) \neq 0_{m\times m}$.

   v. Therefore verify that $\mathrm{tr}(s_{i-1}(A)^2h(A)^2) = \mathrm{tr}((s_{i-1}(A)h(A))^2) = \|s_{i-1}(A)h(A)\|^2 > 0$.

   vi. Therefore verify that $\frac{s_{i-1_0}}{s_{i0}} > 0$.

   vii. **Therefore verify that $s_{i-1_0} > 0$.**

2.47.1 Auxilliary algorithm

Choose an integer $0 \leq k \leq t$ such that polynomial s_k is defined. Choose a polynomial $g = g_0x^k + g_1x^{k-1} + \cdots + g_k$.

1. If $k = t$, then verify that $\mathrm{tr}(g(A)s_k(A)h(A)^2)$

   (a) $= \mathrm{tr}(g(A)s_t(A)h(A)^2)$.

   (b) $= \mathrm{tr}(g(A)0_{m\times m}h(A)^2)$.

   (c) $= \mathrm{tr}(0_{m\times m})$.

   (d) $= 0$.

2. Otherwise if $k = t-1$, then verify that $\mathrm{tr}(g(A)s_k(A)h(A)^2)$

   (a) $= \mathrm{tr}(g(A)s_{t-1}(A)h(A)^2)$.

   (b) $= \mathrm{tr}(g(A)I_{m,m}h(A))$.

   (c) $= \mathrm{tr}(g(A)h(A))$.

   (d) $= g_0$.

   (e) $= \frac{g_0}{s_{k+1_0}}$.

3. Otherwise if $k < t-1$, then do the following:

   (a) Verify that $\mathrm{tr}(g(A)s_k(A)h(A)^2) = \mathrm{tr}(g(A)(q_{k+1}(A)s_{k+1}(A) + s_{k+2}(A))h(A)^2)$.

   (b) Verify that $\mathrm{tr}(g(A)s_k(A)h(A)^2) = \mathrm{tr}(g(A)q_{k+1}(A)s_{k+1}(A)h(A)^2) + \mathrm{tr}(g(A)s_{k+2}(A)h(A)^2)$.

   (c) Execute the auxilliary algorithm supplying the integer $k+1$ and $(k+1)^{th}$ degree polynomial gq.

   (d) Since it should be that $k+1 < t$, the execution of the auxilliary algorithm should have verified that $\mathrm{tr}((g(A)q_{k+1}(A))s_{k+1}(A)h(A)^2) = \frac{(s_{k+2_0}/s_{k+1_0})g_0}{s_{k+2_0}} = \frac{g_0}{s_{k+1}}$.

   (e) Execute the auxilliary algorithm supplying the integer $k+2$ and k^{th} degree polynomial g.

   (f) If $k+2 < t$, then:

      i. The execution of the auxilliary algorithm should have verified that $\mathrm{tr}(g(A)s_{k+2}(A)h(A)^2) = \frac{0}{s_{k+2_0}} = 0$.

   (g) Otherwise if $k+2 = t$, then:

      i. The execution of the auxilliary algorithm should have verified that $\mathrm{tr}(g(A)s_{k+2}(A)h(A)^2) = \mathrm{tr}(g(A)s_t(A)h(A)^2) = 0$.

   (h) **Therefore verify that** $\mathrm{tr}(g(A)s_k(A)h(A)^2) = \frac{g_0}{s_{k+1_0}} + 0 = \frac{g_0}{s_{k+1_0}}$.

2.48 Algorithm 48

Choose an $m \times m$ matrix, A, whose entries are only rationals and is such that $A^T = A$. Now do the following:

1. Execute algorithm 3 on the polynomial matrix $xI - A$ and let D be the result.
2. The last diagonal entry of D should be a product of m factors, $u_1 u_2 \cdots u_m$.
3. Execute algorithm 47 on the matrix A.
4. Execute algorithm 42 supplying the sequences of polynomials $s_0, s_1, \cdots, s_t$ and $q_1, q_2, \cdots, q_{t-1}$ constructed above.
5. For $i = 1$ to $i = t$ do the following:
    (a) If $\operatorname{sgn}(u_1(c_i)) = \operatorname{sgn}(u_1(d_i)), \operatorname{sgn}(u_2(c_i)) = \operatorname{sgn}(u_2(d_i)), \cdots, \operatorname{sgn}(u_m(c_i)) = \operatorname{sgn}(u_m(d_i))$, then do the following:
        i. Verify that $\operatorname{sgn}(u_1(c_i)) \operatorname{sgn}(u_2(c_i)) \cdots \operatorname{sgn}(u_m(c_i)) = \operatorname{sgn}(u_1(d_i)) \operatorname{sgn}(u_2(d_i)) \cdots \operatorname{sgn}(u_m(d_i))$.
        ii. Verify that $\operatorname{sgn}(u_1(c_i) u_2(c_i) \cdots u_m(c_i)) = \operatorname{sgn}(u_1(d_i) u_2(d_i) \cdots u_m(d_i))$.
        iii. Verify that $\operatorname{sgn}(s_t(c_i)) = \operatorname{sgn}(s_t(d_i))$.
        iv. **Abort algorithm.**
    (b) Otherwise do the following:
        i. Assign (c_i, d_i) to one of the u_js for which $\operatorname{sgn}(u_j(c_i)) = -\operatorname{sgn}(u_j(d_i))$.
6. Let n_i be the number of pairs assigned to the polynomial u_i.
7. Verify that $\sum_{i=1}^{m} n_j = t$.
8. If for any $i = 1$ to $i = m$, $n_i > \deg(u_i)$, then do the following:
    (a) Execute algorithm 37 on the polynomial u_i along with $\deg(u_i) + 1$ of the rational number pairs assigned to it.
    (b) **Abort algorithm.**
9. Otherwise if for any $i = 1$ to $i = m$, $n_i < \deg(u_i)$, then do the following:
    (a) Verify that $\sum_{i=1}^{m} n_j < t$.
    (b) Verify that $\sum_{i=1}^{m} n_j < \sum_{i=1}^{m} n_j$.
    (c) **Abort algorithm.**
10. Otherwise if for any $i = 1$ to $i = m$, $n_i \neq \deg(u_i)$, then do the following:
    (a) Verify that $n_i \neq \deg(u_i)$.
    (b) Verify that $n_i \not< \deg(u_i)$.
    (c) Verify that $n_i \not> \deg(u_i)$.
    (d) **Abort algorithm.**
11. **For all $i = 1$ to $i = m$, verify that $n_i = \deg(u_i)$.**

2.49 Algorithm 49 (Upper triangular matrix multiplication)

Choose two upper triangular $m \times m$ matrices, A and B. Now do the following:

1. Multiply A by B and let C be the result.
2. For $i = 1$ to $i = m$, do the following:
    (a) **Verify that** $C_{i,i} = \sum_{k=1}^{m} (A_{i,k} B_{k,i}) = \sum_{k=1}^{i-1} (A_{i,k} B_{k,i}) + A_{i,i} B_{i,i} + \sum_{k=i+1}^{m} (A_{i,k} B_{k,i}) = \sum_{k=1}^{i-1} (0 * B_{k,i}) + A_{i,i} B_{i,i} + \sum_{k=i+1}^{m} (A_{i,k} * 0) = A_{i,i} B_{i,i}$**.**
3. For $i = 2$ to $i = m$, do the following:
    (a) For $j = 1$ to $j = i - 1$, do the following:
        i. Verify that $C_{i,j} = \sum_{k=1}^{m} A_{i,k} B_{k,j} = \sum_{k=1}^{i-1} A_{i,k} B_{k,j} + \sum_{k=i}^{m} A_{i,k} B_{k,j} = \sum_{k=1}^{i-1} 0 * B_{k,j} + \sum_{k=i}^{m} A_{i,k} * 0 = 0$.
4. **Therefore verify that C is upper triangular.**

2.50 Algorithm 50 (Orthogonalization)

Choose integers $m \geq n \geq 0$. Choose an $n \times m$ matrix of polynomials M and an $m \times n$ matrix of polynomials A_0 such that $M A_0 = I_n$. Now do the following:

1. Using algorithm 8, verify that $C_n(M_0 A_0) = C_n(I_n) = 1$.
2. If $C_n(A_0) = 0_{\binom{m}{n} \times 1}$, then do the following:

(a) Verify that $C_n(M_0A_0) = C_n(M_0)C_n(A_0) = C_n(M_0)0_{\binom{m}{n}\times 1} = 0$.

(b) **Abort algorithm.**

3. Otherwise, do the following:

4. Verify that $C_n(A_0) \neq 0_{\binom{m}{n}\times 1}$.

5. For $i = 1$ to $i = n$, do the following:

   (a) If $A_{i-1}e_i = 0_{m\times 1}$, then do the following:

      i. Verify that $C_n(A_{i-1}) = 0$.

      ii. **Abort algorithm.**

   (b) Otherwise, do the following:

   (c) Verify that $\|A_{i-1}e_i\|^2 \neq 0$.

   (d) Let D_i be a $n \times n$ diagonal matrix comprising i 1s followed by $n - i$ $\|A_{i-1}e_i\|^2$s.

   (e) **Verify that D_i is upper triangular.**

   (f) Verify that $C_n(D_i) = (\|A_{i-1}e_i\|^2)^{n-i} \neq 0$.

   (g) Let $N_i = I_n$ except that its i^{th} row is $i - 1$ 0s followed by a 1 followed by $-({A_{i-1}}^TA_{i-1})_{i,i+1}$, then $-({A_{i-1}}^TA_{i-1})_{i,i+2}$, all the way up to $-({A_{i-1}}^TA_{i-1})_{i,n}$.

   (h) **Verify that N_i is upper triangular.**

   (i) Using algorithm 8, verify that $C_n(N_i) = 1 \neq 0$.

   (j) Let $A_i = A_{i-1}D_iN_i$.

   (k) **Verify that $C_n(A_i) = C_n(A_{i-1}D_iN_i) = C_n(A_{i-1})C_n(D_i)C_n(N_i) = C_n(A_{i-1})C_n(D_i) \neq 0$.**

   (l) Verify that ${A_{i-1}}^TA_i = ({A_{i-1}}^TA_{i-1})D_iN_i$ is a matrix with 0s from position $(i, i+1)$ to (i, n).

   (m) **Verify that ${A_i}^TA_i = (A_{i-1}D_iN_i)^T(A_{i-1}D_iN_i) = {N_i}^T{D_i}^T({A_{i-1}}^TA_{i-1})D_iN_i$ is a matrix with 0s from position $(i, i+1)$ to (i, n) and from position $(i+1, i)$ to (n, i).**

   (n) Verify that $A_i = A_0(D_1N_1)\cdots(D_iN_i)$.

   (o) Verify that $MA_i = (D_1N_1)\cdots(D_iN_i)$.

   (p) For $j = 1$ to $j = n$, do the following:

      i. Using algorithm 49, verify that $({e_j}^TM)(A_ie_j) = {e_j}^T(MA_i)e_j = {e_j}^T((D_1N_1)\cdots(D_iN_i))e_j = (D_{1j,j}N_{1j,j})\cdots(D_{ij,j}N_{ij,j})$.

      ii. **Therefore using (5d) verify that $({e_j}^TM)(A_ie_j) = D_{1j,j}\cdots D_{ij,j} = D_{1j,j}\cdots D_{\min(i,j-1)j,j} = \|A_0e_1\|^2\cdots\|A_{\min(i,j-1)-1}e_{\min(i,j-1)}\|^2$.**

6. Let $E = (D_1N_1)\cdots(D_nN_n)$.

7. **Verify that ${A_n}^TA_n = (A_0(D_1N_1)\cdots(D_nN_n))^T(A_0(D_1N_1)\cdots(D_nN_n)) = ((D_1N_1)\cdots(D_nN_n))^T({A_0}^TA_0)((D_1N_1)\cdots(D_nN_n)) = E^T({A_0}^TA_0)E$ is a diagonal matrix.**

8. **Yield the matrix E.**

2.51 Algorithm 51 (Cauchy-Schwarz inequality)

Choose a $1 * m$ matrix A and an $m * 1$ matrix B. Now do the following:

1. Verify that 0

   (a) $\leq \frac{1}{2}\sum_{i=1}^m\sum_{j=1}^m(A_iB_j - A_jB_i)^2$

   (b) $= \frac{1}{2}\sum_{i=1}^m\sum_{j=1}^m({A_i}^2{B_j}^2 - 2A_iB_jA_jB_i + {A_j}^2{B_i}^2)$

   (c) $= \frac{1}{2}\sum_{i=1}^m {A_i}^2\sum_{j=1}^m {B_j}^2 + \frac{1}{2}\sum_{i=1}^m {B_i}^2 \cdot \sum_{j=1}^m {A_j}^2 - \sum_{i=1}^m A_iB_i\sum_{j=1}^m A_jB_j$

   (d) $= \frac{1}{2}(AA^T)(B^TB) + \frac{1}{2}(AA^T)(B^TB) - (AB)^2$

   (e) $= (AA^T)(B^TB) - (AB)^2$.

2. **Therefore verify that $(AB)^2 \leq (AA^T)(B^TB)$.**

2.52 Algorithm 52

Choose integers $m \geq n > 0$. Choose an $n \times m$ matrix of polynomials M and an $m \times n$ matrix of polynomials A_0 such that $MA_0 = I_n$. Choose a rational number x. Now do the following:

1. Execute algorithm 50 on M and A_0.

2. Let $a = \max(\|M(x)\|^2, 1)$.

3. **Choose a column index $1 \leq j \leq n$ such that $\|A_n(x)e_j\|^2 < \frac{1}{a^{(2n+2)!!}}$.**

4. Let $i = n$.

5. Verify that $\|A_i(x)e_j\|^2 < \frac{1}{a^{(2i+2)!!}}$.

6. Using algorithm 51, verify that $(e_j{}^T M(x)A_i(x)e_j)^2 \leq \|e_j{}^T M(x)\|^2\|A_i(x)e_j\|^2 < \|M(x)\|^2\frac{1}{a^{(2i+2)!!}} \leq a\frac{1}{a^{(2i+2)!!}} \leq \frac{1}{a^{(2i)!!*2i}} \leq 1$.

7. If $i = 0$, then do the following:

   (a) Verify that $(e_j{}^T M(x)A_i(x)e_j)^2 = (e_j{}^T M(x)A_0(x)e_j)^2 = (e_j{}^T I_n e_j)^2 = 1$.

   (b) **Abort algorithm.**

8. Otherwise, do the following:

9. Using algorithm 50, verify that $(1\|A_0 e_1\|^2 \cdots \|A_{\min(i,j-1)-1}e_{\min(i,j-1)}\|^2)^2 = (e_j{}^T M(x)A_i(x)e_j)^2 < \frac{1}{a^{(2i)!!*2i}} \leq 1$.

10. If $\min(i, j-1) = 0$, then do the following:

    (a) Verify that $(1\|A_0(x)e_1\|^2 \cdots \|A_{\min(i,j-1)-1}(x)e_{\min(i,j-1)}\|^2)^2 = 1^2 = 1$.

    (b) **Abort algorithm.**

11. Otherwise do the following:

    (a) Verify that $\min(i, j-1) > 0$.

    (b) If for all $k = 0$ to $k = \min(i, j-1) - 1$, $\|A_k(x)e_{k+1}\|^2 \geq \frac{1}{a^{(2i)!!}}$, then do the following:

       i. Verify that $(e_j{}^T M(x)A_i(x)e_j)^2 = (\|A_0(x)e_1\|^2 \cdots \|A_{\min(i,j-1)-1}(x)e_{\min(i,j-1)}\|^2)^2 \geq (\frac{1}{a^{(2i)!!}})^{2\min(i,j-1)} \geq (\frac{1}{a^{(2i)!!}})^{2i} = \frac{1}{a^{(2i)!!*2i}}$.

       ii. **Abort algorithm.**

    (c) Otherwise, do the following:

       i. **Let k, where $0 \leq k < i$, be one of the integers for which $\|A_k(x)e_{k+1}\|^2 < \frac{1}{a^{(2i)!!}}$.**

       ii. **Verify that $\|A_k(x)e_{k+1}\|^2 < \frac{1}{a^{(2i)!!}} \leq \frac{1}{a^{(2k+2)!!}}$.**

       iii. **Simultaneously set i to k and j to $k+1$.**

       iv. **Go to (4).**

12. **Abort algorithm.**

2.53 Algorithm 53

Choose an $m \times m$ matrix, A, whose entries are only rationals and is such that $A^T = A$. Now do the following:

1. Execute algorithm 48 on the matrix A.

2. For $i = 1$ to $i = t$, let k_i be the index of the polynomial to which (c_i, d_i) was associated.

3. Let the macro $[P]$ expand to "(if P, then yield 1, otherwise yield 0)".

4. Verify that $\sum_{i=1}^t (m + 1 - k_i)$

   (a) $= \sum_{i=1}^t \sum_{j=1}^m [k_i \leq j]$

   (b) $= \sum_{j=1}^m \sum_{i=1}^t [k_i \leq j]$

   (c) $= \sum_{j=1}^m \sum_{i=1}^t [k_i \leq j] \sum_{l=1}^m [k_i = l]$

   (d) $= \sum_{j=1}^m \sum_{l=1}^m \sum_{i=1}^t [k_i \leq j][k_i = l]$

   (e) $= \sum_{j=1}^m \sum_{l=1}^m \sum_{i=1}^t [l \leq j][k_i = l]$

   (f) $= \sum_{j=1}^m \sum_{l=1}^m [l \leq j] \sum_{i=1}^t [k_i = l]$

   (g) $= \sum_{j=1}^m \sum_{l=1}^m [l \leq j] n_l$

   (h) $= \sum_{j=1}^m \sum_{l=1}^m [l \leq j] \deg u_l$

   (i) $= \sum_{j=1}^m \sum_{l=1}^j \deg u_l$

   (j) $= \sum_{j=1}^m \deg D_{j,j}$

   (k) $= m$

5. **Verify that $\sum_{i=1}^t (m + 1 - k_i) = m$.**

2.54 Algorithm 54 (Spectral algorithm initialization)

Choose an $m \times m$ matrix, A, whose entries are only rationals and is such that $A^T = A$. Choose a rational number $\epsilon > 0$. Now do the following:

1. Execute algorithm 48 on the matrix A.

2. Execute algorithm 6 with $xI_m - A$ as the choice matrix. Take note of M^{-1}, D, and N^{-1}.

3. Let M' be the matrix obtained by replacing all the negative signs in M^{-1} with positive signs.

4. Let $M'' = \max_{i=1}^m \max_{j=1}^m M'(\max(|c_1|, |d_t|))_{i,j}$.

5. Let N' be the matrix obtained by replacing all the negative signs in N with positive signs.

6. Let $N'' = 1 + \max_{i=1}^m \max_{j=1}^m N'(\max(|c_1|, |d_t|))_{i,j}$.
7. Let L be the formal polynomial obtained by replacing all the negative signs in $(\|N^{-1}\|^2)^{(2m+2)!!}$ with positive signs.
8. Let $L' = \frac{1}{\max(1, L(|c_1|), L(|d_t|))}$.
9. Let $\delta = \min(1, \min_{i=1}^{t-1}(c_{i+1} - c_i))$.
10. For $i = 1$ to $i = t$, do the following:
    (a) Let k_i be the index of the polynomial to which (c_i, d_i) was associated.
    (b) Verify that $\text{sgn}(u_{k_i}(c_i)) \neq \text{sgn}(u_{k_i}(d_i))$.
    (c) Let Q be the last $m+1-k_i$ columns of I_m.
    (d) Execute algorithm 50 on the matrix NQ. Let E be the $(m+1-k_i) \times (m+1-k_i)$ matrix yielded from this.
    (e) Let $K_i = NQE$, an $m \times (m+1-k_i)$ matrix.
    (f) **Verify that $K_i{}^T K_i$ is a diagonal matrix.**
    (g) Let E' be the matrix obtained by replacing all the negative signs in E with positive signs.
    (h) Let $E_i'' = \max_{j=1}^m \max_{l=1}^m E'(\max(|c_1|, |d_t|))_{j,l}$.
11. Let $E'' = 1 + \max_{i=1}^t E_i''$.
12. For $i = 1$ to $i = t$, do the following with the symbols Q, E, and F retaining their values from the corresponding iteration of the loop at (7).
    (a) Let $b = \frac{\epsilon\delta}{M''N''E''^2 m^2(m+1-k_i)}$.
    (b) For $j = k$ to $j = m$, do the following:
        i. Execute algorithm 36 on the formal polynomial $D_{j,j}$, interval (c_i, d_i), and target of b. Let c_i and d_i receive their updates.
    (c) If a diagonal entry of $K_i(c_i)^T K_i(c_i)$ is less than L', then do the following:
        i. Let z be the index of the column of the entry.
        ii. Verify that $(Q^T N^{-1})(NQ) = Q^T(N^{-1}N)Q = Q^T I_m Q = Q^T Q = I_{m+1-k_i}$.
        iii. Verify that $L' \leq \frac{1}{\max(\|(Q^T N^{-1})(c_i)\|^2, 1)^{(2(m+1-k_i)+2)!!}}$.
        iv. Execute algorithm 52 with matrices $Q^T N^{-1}$ and NQ, rational number c_i, and column index z.
        v. **Abort algorithm.**
    (d) Otherwise, do the following:
        i. **For $j = 1$ to $j = m+1-k_i$, verify that $(K_i(c_i)^T K_i(c_i))_{j,j} \geq L'$.**
        ii. Verify that $xK_i - AK_i = (xI_m - A)K_i = M^{-1}DN^{-1}K_i = M^{-1}DN^{-1}NQE = M^{-1}DQE$.
        iii. Verify that $\|c_i K_i(c_i) - AK_i(c_i)\|^2 = \|M^{-1}(c_i)D(c_i)QE(c_i)\|^2 \leq \|M'' J_m \frac{\epsilon\delta}{M''N''E''^2 m^2(m+1-k_i)} QE'' J_{m+1-k}\|^2 = \|J_m \frac{\epsilon\delta}{N''E''m^2(m+1-k_i)} QJ_{m+1-k}\|^2 = \|\frac{\epsilon\delta}{N''E''m^2} J_{m\times(m+1-k_i)}\|^2 \leq \|\frac{\epsilon\delta}{N''E''m^2} J_{m\times(m+1-k_i)}\|^2 = \frac{m+1-k_i}{m^3} \cdot \frac{\epsilon^2\delta^2}{(N''E'')^2}$.
        iv. **Therefore verify that $\|c_i K_i(c_i) - AK_i(c_i)\|^2 \leq \frac{m+1-k_i}{m^3} \cdot \frac{\epsilon^2\delta^2}{(N''E'')^2} \leq \frac{m+1-k_i}{m}\epsilon^2$.**

2.55 Algorithm 55 (Spectral algorithm)

Choose an $m \times m$ matrix, A, whose entries are only rationals and is such that $A^T = A$. Choose a rational number $\epsilon > 0$. Now do the following:

1. Execute algorithm 54 on matrix A and rational ϵ.
2. Let C be a diagonal matrix whose i^{th}, where $1 \leq i \leq t$, group of entries are $m+1-k_i$ c_is.
3. Using algorithm 53, verify that C is $m \times m$.
4. Let K be a matrix whose columns are the in-order concatenation of those of $K_1(c_1), K_2(c_2), \cdots, K_t(c_t)$.
5. Using algorithm 53 and (9e), verify that K is $m \times m$.
6. **Using algorithm 53 and algorithm 54, verify that $\|KC - AK\|^2 \leq \sum_{i=1}^t \frac{m+1-k_i}{m}\epsilon^2 = \frac{\sum_{i=1}^t (m+1-k_i)}{m}\epsilon^2 = \frac{m}{m}\epsilon^2 = \epsilon^2$.**

7. For $i = 1$ to $i = m$, do the following: For $j = 1$ to $j = m$, do the following:
   (a) If Ke_i was constructed during iteration $i = a$ and Ke_j during iteration $i = b$ of (11), and if $a \neq b$, then do the following:
   (b) Verify that $|(c_b - c_a)(Ke_i)^T(Ke_j)|$
      i. $= |c_b(Ke_i)^T(Ke_j) - c_a(Ke_i)^T(Ke_j)|$
      ii. $= |(Ke_i)^T(c_bKe_j) - (c_aKe_i)^T(Ke_j)|$
      iii. $= |(Ke_i)^T(AKe_j + c_bKe_j - AKe_j) - (AKe_i + c_aKe_i - AKe_i)^T(Ke_j)|$
      iv. $\leq |(Ke_i)^T(AKe_j) - (AKe_i)^T(Ke_j)| + |(Ke_i)^T(c_bKe_j - AKe_j)| + |(c_aKe_i - AKe_i)^T(Ke_j)|$
      v. $\leq |(Ke_i)^T A(Ke_j) - (Ke_i)^T A^T(Ke_j)| + |mN''E''J_{1\times m}\frac{\epsilon\delta}{N''E''m^2}J_{m\times 1}| + |\frac{\epsilon\delta}{N''E''m^2}J_{1\times m}mN''E''J_{m\times 1}|$
      vi. $= 2\epsilon\delta$.
   (c) **Therefore verify that $|{e_i}^T(K^TK)e_j| = |(Ke_i)^T(Ke_j)| \leq \frac{2\epsilon\delta}{c_b - c_a} \leq 2\epsilon$.**
8. **Using (7c) and algorithm 54, verify that the absolute values of all the non-diagonal entries K^TK are less than or equal to 2ϵ.**
9. **Using algorithm 54, verify that all the diagonal entries of K^TK are more than or equal to L'.**