Various New Methods for Computing Subresultant Polynomial Remainder Sequences (PRS's)

Alkiviadis G. Akritas

University of Thessaly
Department of Electrical and Computer Engineering
Volos, Greece

July 21, 2015

Problem 1/2

Problem 1/2

- See https://en.wikipedia.org/wiki/Polynomial_greatest_common_divisor#Pseudo-remainder_sequences.

Problem 2/2

Problem 2/2

- Also, note the caveat in http://planetmath.org/sturmstheorem

...

Problem 2/2

- Also, note the caveat in http://planetmath.org/sturmstheorem ...

- "Be aware that some computer algebra systems may normalize remainders from the Euclidean Algorithm which messes up the sign."

Problem 2/2

- Also, note the caveat in http://planetmath.org/sturmstheorem ...

- "Be aware that some computer algebra systems may normalize remainders from the Euclidean Algorithm which messes up the sign."

- The culprit is ...

Problem 2/2

- Also, note the caveat in http://planetmath.org/sturmstheorem ...

- "Be aware that some computer algebra systems may normalize remainders from the Euclidean Algorithm which messes up the sign."

- The culprit is ...

- ... the function `prem(f, g, x)` for computing pseudo-remainders!!

Outline of the talk 1/2

Outline of the talk 1/2

- We first present Sylvester's matrices of 1840 and 1853 and show how they can be used to compute, in $\mathbb{Z}[x]$, Euclidean subresultant prs's and Sturmian modified subresultant sequences (or modified subresultant prs), respectively.

Outline of the talk 1/2

- We first present Sylvester's matrices of 1840 and 1853 and show how they can be used to compute, in $\mathbb{Z}[x]$, Euclidean subresultant prs's and Sturmian modified subresultant sequences (or modified subresultant prs), respectively.
- We next review the Collins-Brown-Traub method for computing subresultant polynomial remainder sequences (prs's) in $\mathbb{Z}[x]$ using pseudo-divisions. We will use the `sympy` pseudo-remainder function `prem(f, g, x)` and point out the drawback of this approach.

Outline of the talk 2/2

Outline of the talk 2/2

- Finally, we present the Pell-Gordon theorem of 1917, which motivated us to develop the following new methods for computing subresultant prs's:

Outline of the talk 2/2

- Finally, we present the Pell-Gordon theorem of 1917, which motivated us to develop the following new methods for computing subresultant prs's:
- **1st method**, which uses a new function, `remZ`, for computing remainders in $\mathbb{Z}[x]$.

Outline of the talk 2/2

- Finally, we present the Pell-Gordon theorem of 1917, which motivated us to develop the following new methods for computing subresultant prs's:
- **1st method**, which uses a new function, `remZ`, for computing remainders in $\mathbb{Z}[x]$.
- **2nd method**, which performs rational divisions, i.e. polynomial divisions are performed in $\mathbb{Q}[x]$ but the final result is in $\mathbb{Z}[x]$.

Outline of the talk 2/2

- Finally, we present the Pell-Gordon theorem of 1917, which motivated us to develop the following new methods for computing subresultant prs's:
- **1st method**, which uses a new function, `remZ`, for computing remainders in $\mathbb{Z}[x]$.
- **2nd method**, which performs rational divisions, i.e. polynomial divisions are performed in $\mathbb{Q}[x]$ but the final result is in $\mathbb{Z}[x]$.
- **3nd method**, which triangularizes the Sylvester matrix of 1853 in $\mathbb{Z}[x]$, while using the Sylvester matrix of 1840 to compute the correct sign and value of the polynomial coefficients.

Table of contents

Consider the polynomial $f, g \in \mathbb{Z}[x]$ of degrees n, m respectively, with $n \geq m$.

Consider the polynomial $f, g \in \mathbb{Z}[x]$ of degrees n, m respectively, with $n \geq m$.

We call Sylvester's matrix of 1840 `sylvester1`. Its dimensions are $(n+m) \times (n+m)$ and it consists of two groups of rows: the first one with m rows and the second one with n. Note that ...

Consider the polynomial $f, g \in \mathbb{Z}[x]$ of degrees n, m respectively, with $n \geq m$.

We call Sylvester's matrix of 1840 `sylvester1`. Its dimensions are $(n+m) \times (n+m)$ and it consists of two groups of rows: the first one with m rows and the second one with n. Note that ...

$$\det(sylvester1(f, g, x)) = resultant(f, g, x)$$

Consider the polynomial $f, g \in \mathbb{Z}[x]$ of degrees n, m respectively, with $n \geq m$.

We call Sylvester's matrix of 1840 `sylvester1`. Its dimensions are $(n+m) \times (n+m)$ and it consists of two groups of rows: the first one with m rows and the second one with n. Note that ...

$$\det(sylvester1(f, g, x)) = resultant(f, g, x)$$

We call Sylvester's matrix of 1853 `sylvester2`. Its dimensions are $2n \times 2n$ and it consists of n pairs of rows. Note that ...

Consider the polynomial $f, g \in \mathbb{Z}[x]$ of degrees n, m respectively, with $n \geq m$.

We call Sylvester's matrix of 1840 `sylvester1`. Its dimensions are $(n+m) \times (n+m)$ and it consists of two groups of rows: the first one with m rows and the second one with n. Note that ...

$$\det(sylvester1(f, g, x)) = resultant(f, g, x)$$

We call Sylvester's matrix of 1853 `sylvester2`. Its dimensions are $2n \times 2n$ and it consists of n pairs of rows. Note that ...

$$\det(sylvester2(f, g, x)) = modified\_resultant(f, g, x)$$

Sylvester's matrix `sylvester1` (1840)

Sylvester's matrix `sylvester1` (1840)

Matrix `sylvester1` consists of two groups of rows ...

Sylvester's matrix `sylvester1` (1840)

Matrix `sylvester1` consists of two groups of rows ...

In the first row of the first group (of m rows) are the coefficients of $f(x)$ with $m-1$ trailing zeros. The second row in this group differs from the first one in that its elements have been rotated to the right by one. A total of $m-1$ rotations are needed to construct the first group of rows.

Sylvester's matrix `sylvester1` (1840)

Matrix `sylvester1` consists of two groups of rows ...

In the first row of the first group (of m rows) are the coefficients of $f(x)$ with $m-1$ trailing zeros. The second row in this group differs from the first one in that its elements have been rotated to the right by one. A total of $m-1$ rotations are needed to construct the first group of rows.

In the first row of the second group (of n rows) are the coefficients of $g(x)$ with $n-1$ trailing zeros. The second row in this group differs from the first one in that its elements have been rotated to the right by one. A total of $n-1$ rotations are needed to construct the second group of rows.

Example of sylvester1

Example of `sylvester1`

Consider the polynomial

$$f = x^3 - 7x + 7$$

and its derivative $g = 3x^2 - 7$:

Example of sylvester1

Consider the polynomial

$$f = x^3 - 7x + 7$$

and its derivative $g = 3x^2 - 7$:

The sylvester1 matrix of f, g is

$$\begin{pmatrix} 1 & 0 & -7 & 7 & 0 \\ 0 & 1 & 0 & -7 & 7 \\ 3 & 0 & -7 & 0 & 0 \\ 0 & 3 & 0 & -7 & 0 \\ 0 & 0 & 3 & 0 & -7 \end{pmatrix}.$$

Sylvester's matrix `sylvester2` (1853)

Sylvester's matrix `sylvester2` (1853)

Matrix `sylvester2` consists of n pairs of rows ...

Sylvester's matrix `sylvester2` (1853)

Matrix `sylvester2` consists of n pairs of rows ...

In the first row of the first pair are the coefficients of $f(x)$ whereas in the second row of the first pair are the coefficients of $g(x)$; $n - m$ zeros have been *pre*pended to $g(x)$ to also make it of degree n.

Sylvester's matrix `sylvester2` (1853)

Matrix `sylvester2` consists of n pairs of rows ...

In the first row of the first pair are the coefficients of $f(x)$ whereas in the second row of the first pair are the coefficients of $g(x)$; $n - m$ zeros have been *pre*pended to $g(x)$ to also make it of degree n.

Both rows in the first pair have $2n - (n + 1)$ trailing zeros and both rows of the last pair have $2n - (n + 1)$ leading zeros. The second pair of rows differs from the first one in that the elements of both rows have been rotated to the right by one. A total of $2n - (n + 1)$ rotations are needed to construct `sylvester2`.

Example of sylvester2

Example of `sylvester2`

Consider the polynomial

$$f = x^3 - 7x + 7$$

and its derivative $g = 3x^2 - 7$:

Example of sylvester2

Consider the polynomial

$$f = x^3 - 7x + 7$$

and its derivative $g = 3x^2 - 7$:

The sylvester2 matrix of f, g is

$$\begin{pmatrix} 1 & 0 & -7 & 7 & 0 & 0 \\ 0 & 3 & 0 & -7 & 0 & 0 \\ 0 & 1 & 0 & -7 & 7 & 0 \\ 0 & 0 & 3 & 0 & -7 & 0 \\ 0 & 0 & 1 & 0 & -7 & 7 \\ 0 & 0 & 0 & 3 & 0 & -7 \end{pmatrix}.$$

$\det(sylvester1(f, g, x))$ *versus* $det(sylvester2(f, g, x))$

$\det(sylvester1(f, g, x))$ *versus* $det(sylvester2(f, g, x))$

In general we have the inequality ...

$\det(sylvester1(f, g, x))$ *versus* $det(sylvester2(f, g, x))$

In general we have the inequality ...

$$\det(sylvester1(f, g, x)) \neq det(sylvester2(f, g, x))$$

$\det(sylvester1(f, g, x))$ *versus* $det(sylvester2(f, g, x))$

In general we have the inequality ...

$$\det(sylvester1(f, g, x)) \neq det(sylvester2(f, g, x))$$

For example, given the same polynomials $f = x^3 - 7x + 7$ and $g = 3x^2 - 7$, we have:

$\det(sylvester1(f, g, x))$ *versus* $det(sylvester2(f, g, x))$

In general we have the inequality ...

$$\det(sylvester1(f, g, x)) \neq det(sylvester2(f, g, x))$$

For example, given the same polynomials $f = x^3 - 7x + 7$ and $g = 3x^2 - 7$, we have:

- $\det(sylvester1(f, g, x)) = -49$, whereas,

$\det(sylvester1(f, g, x))$ *versus* $det(sylvester2(f, g, x))$

In general we have the inequality ...

$$\det(sylvester1(f, g, x)) \neq det(sylvester2(f, g, x))$$

For example, given the same polynomials $f = x^3 - 7x + 7$ and $g = 3x^2 - 7$, we have:

- $\det(sylvester1(f, g, x)) = -49$, whereas,
- $\det(sylvester2(f, g, x)) = 49$.

Preliminaries

Polynomial remainder sequences are obtained from ...

Preliminaries

Polynomial remainder sequences are obtained from ...

- ... Euclid's algorithm for polynomial gcd, and

Preliminaries

Polynomial remainder sequences are obtained from ...

- ... Euclid's algorithm for polynomial gcd, and

- ... Sturm's algorithm, which actually is a modification of the Euclidean algorithm. (Since, at each step, we take the negative of the remainder.)

Preliminaries

Polynomial remainder sequences are obtained from ...

- ... Euclid's algorithm for polynomial gcd, and

- ... Sturm's algorithm, which actually is a modification of the Euclidean algorithm. (Since, at each step, we take the negative of the remainder.)

If there are no gaps in the sequence of the degrees of the polynomials in the prs (i.e. no missing remainder), then the prs is called *complete*, else it is called *incomplete*.

Sign sequence of a prs

Sign sequence of a prs

For the sequences of polynomial remainders examined in this talk the following definition is needed:

Sign sequence of a prs

For the sequences of polynomial remainders examined in this talk the following definition is needed:

Definition

The *sign sequence* of a polynomial remainder sequence (prs) is the sequence of signs of the leading coefficients of its polynomials.

Subresultant prs

Subresultant prs

Given $f(x), g(x) \in \mathbb{Z}[x]$ of degrees $\deg(f) = n$ and $\deg(g) = m$ with $n \geq m$ their (proper) *subresultant* prs is a sequence of polynomials similar to the *Euclidean* prs, the sequence obtained by applying in $\mathbb{Q}[x]$ Euclid's polynomial gcd algorithm on $f(x), g(x)$.

Subresultant prs

Given $f(x), g(x) \in \mathbb{Z}[x]$ of degrees $\deg(f) = n$ and $\deg(g) = m$ with $n \geq m$ their (proper) *subresultant* prs is a sequence of polynomials similar to the *Euclidean* prs, the sequence obtained by applying in $\mathbb{Q}[x]$ Euclid's polynomial gcd algorithm on $f(x), g(x)$.

The two sequences differ in that the coefficients of each polynomial in the subresultant prs are the determinants, or *subresultants*, of sub-matrices of `sylvester1`.

Subresultant prs

Given $f(x), g(x) \in \mathbb{Z}[x]$ of degrees $\deg(f) = n$ and $\deg(g) = m$ with $n \geq m$ their (proper) *subresultant* prs is a sequence of polynomials similar to the *Euclidean* prs, the sequence obtained by applying in $\mathbb{Q}[x]$ Euclid's polynomial gcd algorithm on $f(x), g(x)$.

The two sequences differ in that the coefficients of each polynomial in the subresultant prs are the determinants, or *subresultants*, of sub-matrices of `sylvester1`.

The sign sequences of the Euclidean and subresultant prs's are identical for complete prs's. They may differ for incomplete prs's.

Modified subresultant prs

Modified subresultant prs

Given $f(x), g(x) \in \mathbb{Z}[x]$ of degrees $\deg(f) = n$ and $\deg(g) = n - 1$, with $g = f'$, their *modified subresultant* prs is a sequence of polynomials similar to the sequence obtained by applying in $\mathbb{Q}[x]$ Sturm's algorithm on $f(x), g(x)$.

Modified subresultant prs

Given $f(x), g(x) \in \mathbb{Z}[x]$ of degrees $\deg(f) = n$ and $\deg(g) = n - 1$, with $g = f'$, their *modified subresultant* prs is a sequence of polynomials similar to the sequence obtained by applying in $\mathbb{Q}[x]$ Sturm's algorithm on $f(x), g(x)$.

The two sequences differ in that the coefficients of each polynomial in the modified subresultant prs are the determinants, or *modified subresultants*, of sub-matrices of `sylvester2`.

Modified subresultant prs

Given $f(x), g(x) \in \mathbb{Z}[x]$ of degrees $\deg(f) = n$ and $\deg(g) = n - 1$, with $g = f'$, their *modified subresultant* prs is a sequence of polynomials similar to the sequence obtained by applying in $\mathbb{Q}[x]$ Sturm's algorithm on $f(x), g(x)$.

The two sequences differ in that the coefficients of each polynomial in the modified subresultant prs are the determinants, or *modified subresultants*, of sub-matrices of `sylvester2`.

The sign sequences in Sturm's and modified subresultant prs's are identical for complete prs's. They may differ for incomplete prs's.

Remarks on proper or modified subresultant prs's (1/2)

Remarks on proper or modified subresultant prs's (1/2)

▶ Sylvester had proved that for complete sequences, the proper or modified subresultant prs is identical to the Euclidean or Sturmian prs, respectively (computed in $\mathbb{Z}[x]$), where the polynomial remainder p_{i+2} in the latter sequences has been divided by the square of the leading coefficient of the polynomial p_i.

Remarks on proper or modified subresultant prs's (1/2)

▶ Sylvester had proved that for complete sequences, the proper or modified subresultant prs is identical to the Euclidean or Sturmian prs, respectively (computed in $\mathbb{Z}[x]$), where the polynomial remainder p_{i+2} in the latter sequences has been divided by the square of the leading coefficient of the polynomial p_i.

▶ Moreover, Sylvester pointed out, that the coefficients of the polynomial remainders obtained as subresultants are the *smallest possible* without introducing rationals and without computing (integer) greatest common divisors.

Remarks on proper or modified subresultant prs's (2/2)

Remarks on proper or modified subresultant prs's (2/2)

- For incomplete prs's, Pell and Gordon presented in 1917 a theorem, relating the polynomials in a Sturmian prs (computed with polynomial divisions in $\mathbb{Q}[x]$) with the corresponding ones of the modified subresultant prs — obtained in $\mathbb{Z}[x]$ by evaluating determinants of submatrices of `sylvester2`.

Remarks on proper or modified subresultant prs's (2/2)

- For incomplete prs's, Pell and Gordon presented in 1917 a theorem, relating the polynomials in a Sturmian prs (computed with polynomial divisions in $\mathbb{Q}[x]$) with the corresponding ones of the modified subresultant prs — obtained in $\mathbb{Z}[x]$ by evaluating determinants of submatrices of `sylvester2`.

- The corresponding result for incomplete Euclidean prs's and `sylvester1` was discovered 30 and 50 years latter by Habicht (1947) and Collins, Brown, Traub, ($\simeq$ 1970) respectively.

Example of a complete subresultant prs

Example of a complete subresultant prs

Consider the polynomial $f = x^3 - 7x + 7$ and its derivative $g = 3x^2 - 7$.

Example of a complete subresultant prs

Consider the polynomial $f = x^3 - 7x + 7$ and its derivative $g = 3x^2 - 7$.

Using the function `subresultants` of `sympy` we see that the complete subresultant prs of f, g is

Example of a complete subresultant prs

Consider the polynomial $f = x^3 - 7x + 7$ and its derivative $g = 3x^2 - 7$.

Using the function `subresultants` of `sympy` we see that the complete subresultant prs of f, g is

$$x^3 - 7x + 7, \quad 3x^2 - 7, \quad -42x + 63, \quad -49.$$

Example of a complete subresultant prs

Consider the polynomial $f = x^3 - 7x + 7$ and its derivative $g = 3x^2 - 7$.

Using the function `subresultants` of `sympy` we see that the complete subresultant prs of f, g is

$$x^3 - 7x + 7, \quad 3x^2 - 7, \quad -42x + 63, \quad -49.$$

The coefficients of $-42x + 63$ are determinants of submatrices of `sylvester1`, whereas for the last member of the sequence we have $-49 = \det(sylvester1(f, g, x))$

Example of a complete modified subresultant prs

Example of a complete modified subresultant prs

Consider again the polynomial $f = x^3 - 7x + 7$ and its derivative $g = 3x^2 - 7$.

Example of a complete modified subresultant prs

Consider again the polynomial $f = x^3 - 7x + 7$ and its derivative $g = 3x^2 - 7$.

Using our own function `subresultants_PG`, written for `sympy`, we see that the complete modified subresultant prs of f, g is

Example of a complete modified subresultant prs

Consider again the polynomial $f = x^3 - 7x + 7$ and its derivative $g = 3x^2 - 7$.

Using our own function `subresultants_PG`, written for `sympy`, we see that the complete modified subresultant prs of f, g is

$$x^3 - 7x + 7, \quad 3x^2 - 7, \quad 42x - 63, \quad 49.$$

Example of a complete modified subresultant prs

Consider again the polynomial $f = x^3 - 7x + 7$ and its derivative $g = 3x^2 - 7$.

Using our own function `subresultants_PG`, written for `sympy`, we see that the complete modified subresultant prs of f, g is

$$x^3 - 7x + 7, \quad 3x^2 - 7, \quad 42x - 63, \quad 49.$$

The coefficients of $42x - 63$ are determinants of submatrices of `sylvester2`, whereas for the last member of the sequence we have $49 = \det(sylvester2(f, g, x))$

In summary ... 1/4

In summary ... 1/4

- Euclidean subresultant prs's are related to matrix `sylvester1`, whereas, ...

In summary ... 1/4

- Euclidean subresultant prs's are related to matrix `sylvester1`, whereas, ...

- Sturmian (modified) subresultant prs's are related to matrix `sylvester2`.

In summary ... 1/4

- Euclidean subresultant prs's are related to matrix `sylvester1`, whereas, ...

- Sturmian (modified) subresultant prs's are related to matrix `sylvester2`.

- Whether computed in $\mathbb{Q}[x]$ or in $\mathbb{Z}[x]$, the signs of Sturmian (modified) prs's must always be the same.

In summary ... 1/4

- Euclidean subresultant prs's are related to matrix `sylvester1`, whereas, ...

- Sturmian (modified) subresultant prs's are related to matrix `sylvester2`.

- Whether computed in $\mathbb{Q}[x]$ or in $\mathbb{Z}[x]$, the signs of Sturmian (modified) prs's must always be the same.

- By contrast, the signs of Euclidean prs's computed in $\mathbb{Q}[x]$ have been allowed to differ from the corresponding ones computed in $\mathbb{Z}[x]$.

In summary ... 1/4

- Euclidean subresultant prs's are related to matrix `sylvester1`, whereas, ...

- Sturmian (modified) subresultant prs's are related to matrix `sylvester2`.

- Whether computed in $\mathbb{Q}[x]$ or in $\mathbb{Z}[x]$, the signs of Sturmian (modified) prs's must always be the same.

- By contrast, the signs of Euclidean prs's computed in $\mathbb{Q}[x]$ have been allowed to differ from the corresponding ones computed in $\mathbb{Z}[x]$.

- See `https://en.wikipedia.org/wiki/Polynomial_greatest_common_divisor#Pseudo-remainder_sequences`.

In summary ... 2/4

In summary ... 2/4

- In `http://planetmath.org/sturmstheorem` it is stated:

In summary ... 2/4

- In `http://planetmath.org/sturmstheorem` it is stated:

- "Be aware that some computer algebra systems may normalize remainders from the Euclidean Algorithm which messes up the sign."

In summary ... 2/4

- In `http://planetmath.org/sturmstheorem` it is stated:

- "Be aware that some computer algebra systems may normalize remainders from the Euclidean Algorithm which messes up the sign."

- Why is that?? Who is the culprit?

In summary ... 2/4

- In `http://planetmath.org/sturmstheorem` it is stated:

- "Be aware that some computer algebra systems may normalize remainders from the Euclidean Algorithm which messes up the sign."

- Why is that?? Who is the culprit?

- The culprit is the function prem, that computes the pseudo-remainder. As explained below, we completely avoid it in our new methods.

In summary ... 3/4

Euclidean PRS

Sturmian PRS

In summary ... 3/4

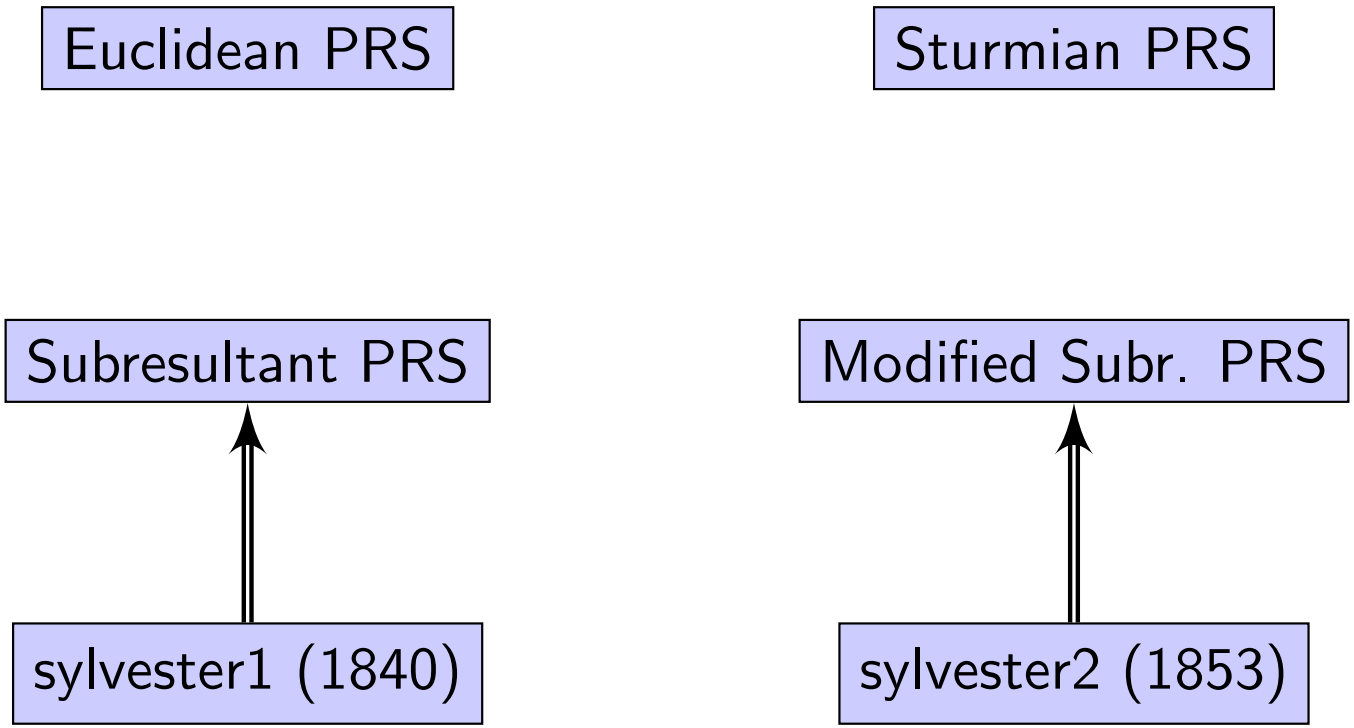
Euclidean PRS

Sturmian PRS

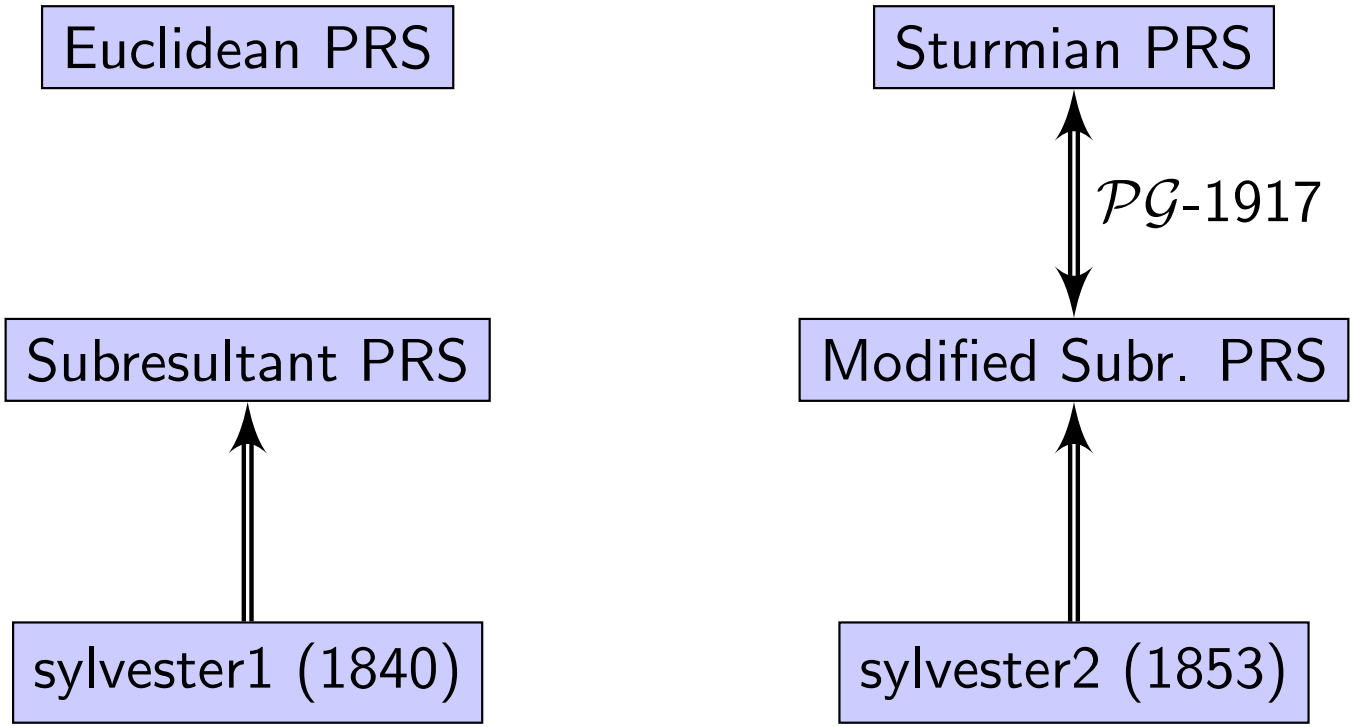
Subresultant PRS

Modified Subr. PRS

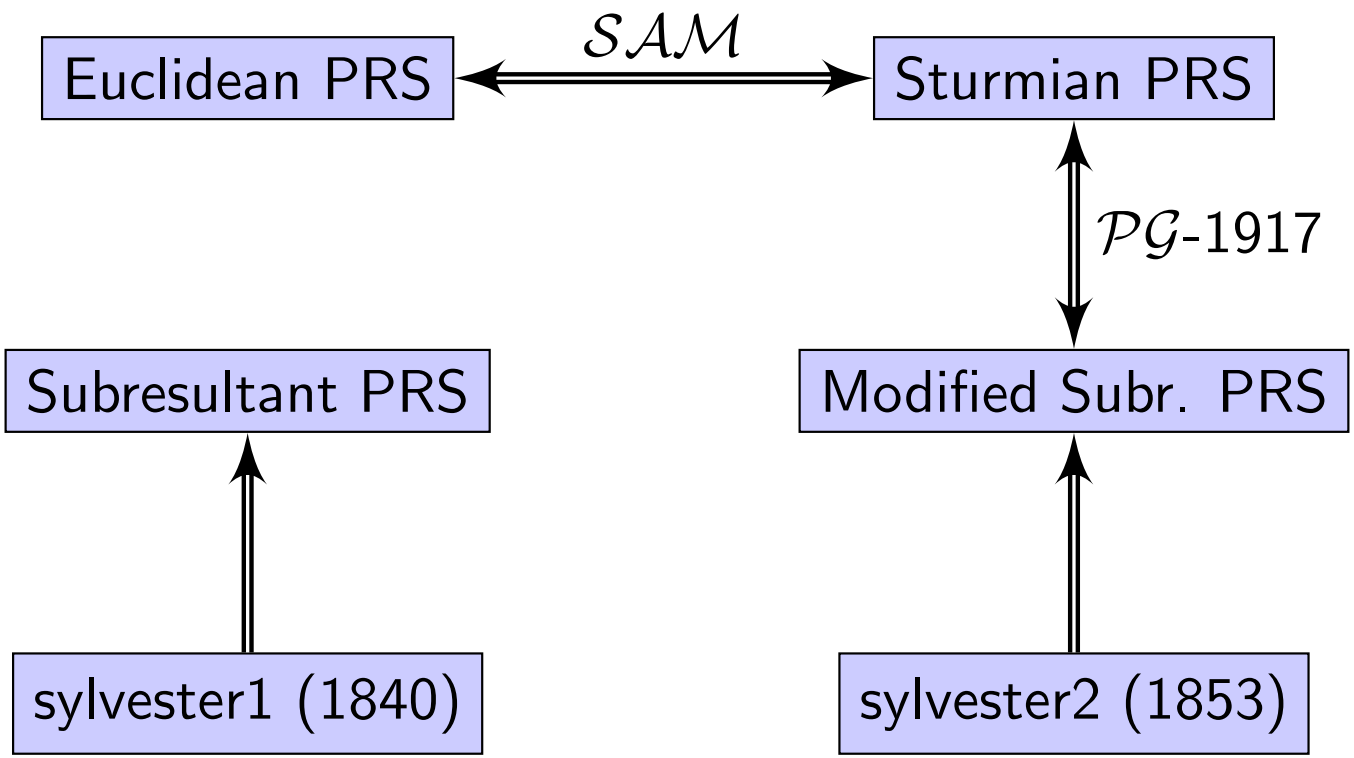
In summary ... 3/4



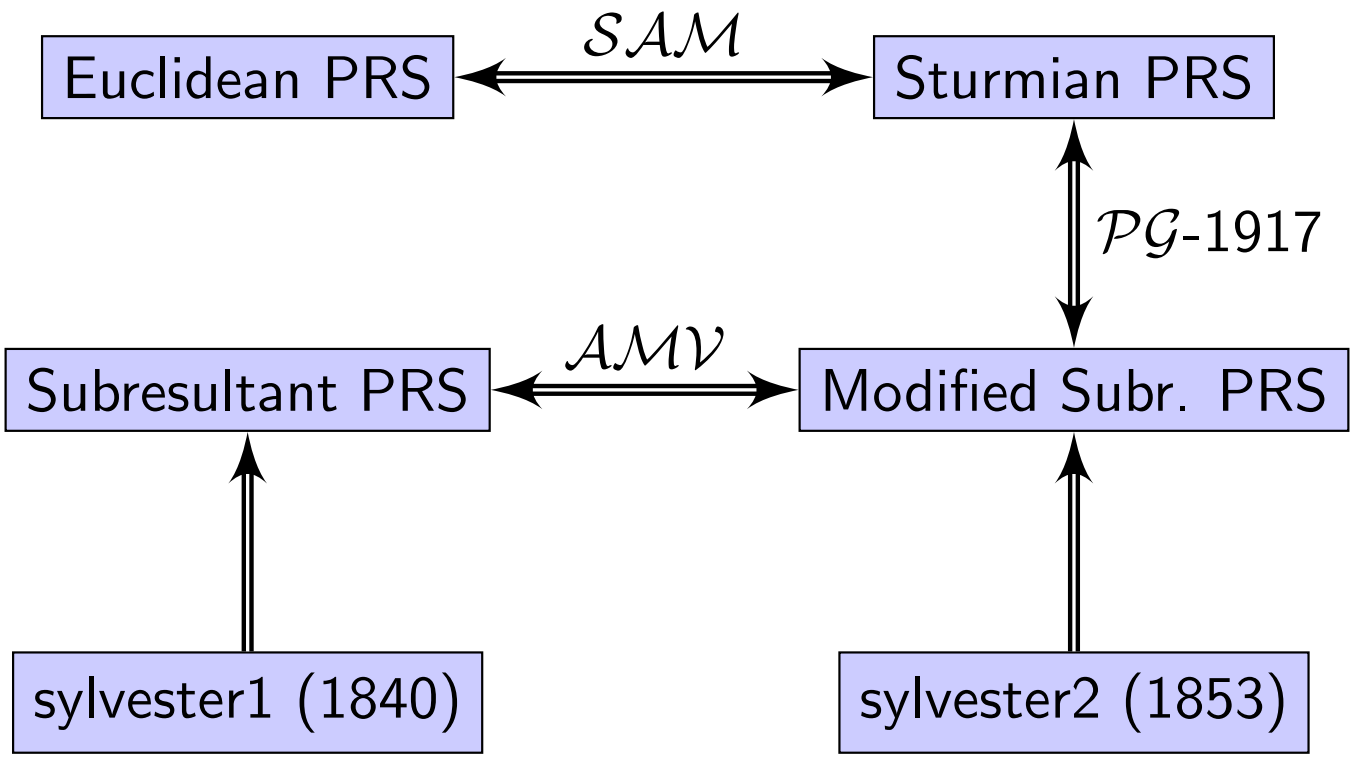
In summary ... 3/4



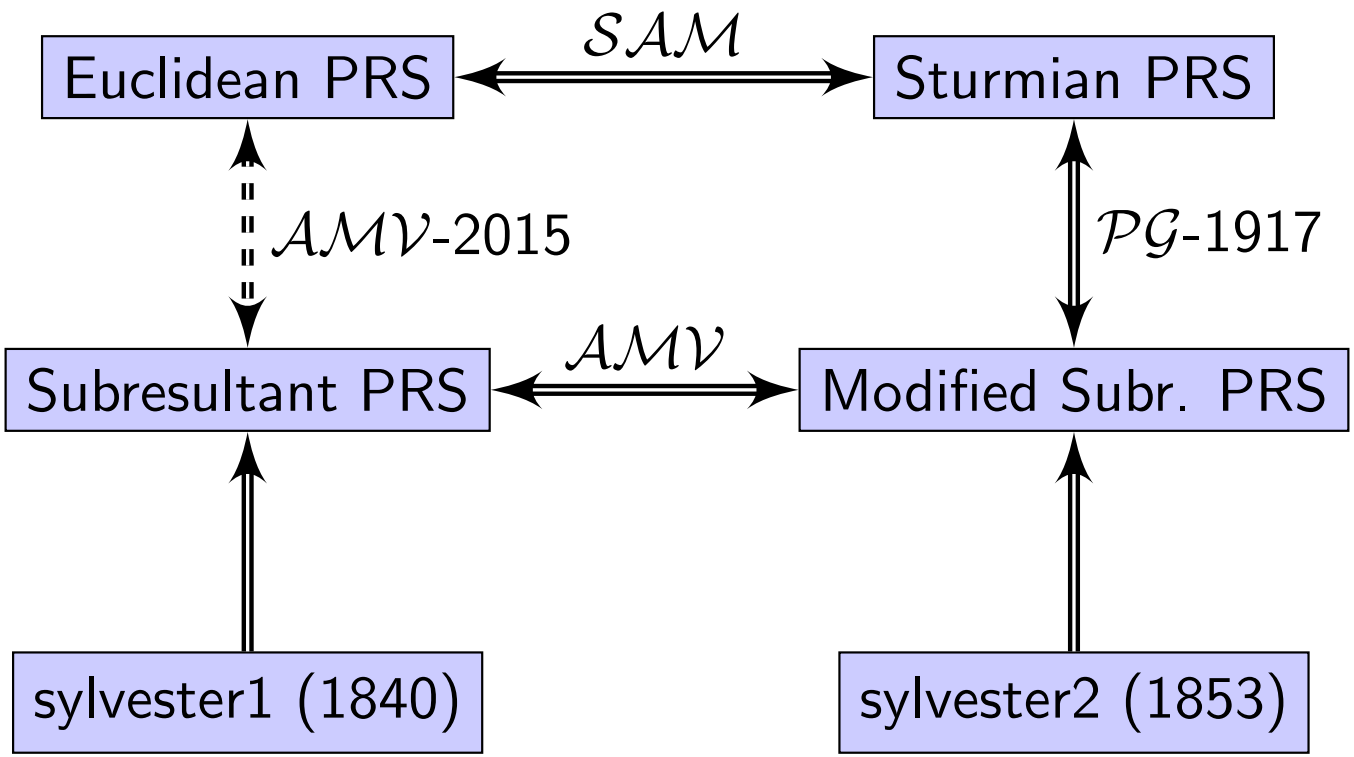
In summary ... 3/4



In summary ... 3/4



In summary ... 3/4



In summary ... 4/4

In summary ... 4/4

- According to the previous diagram, there is a one-to-one correspondence between the subresultant prs and the Euclidean prs.

In summary ... 4/4

- According to the previous diagram, there is a one-to-one correspondence between the subresultant prs and the Euclidean prs.

- The problem with `prem(f, g, x)` is that whereas ...

In summary ... 4/4

- According to the previous diagram, there is a one-to-one correspondence between the subresultant prs and the Euclidean prs.
- The problem with `prem(f, g, x)` is that whereas ...
- ... it correctly computes the subresultant prs,

In summary ... 4/4

- According to the previous diagram, there is a one-to-one correspondence between the subresultant prs and the Euclidean prs.
- The problem with `prem(f, g, x)` is that whereas ...
- ... it correctly computes the subresultant prs,
- ... it computes the wrong signs of the Euclidean prs.

In summary ... 4/4

- According to the previous diagram, there is a one-to-one correspondence between the subresultant prs and the Euclidean prs.
- The problem with `prem(f, g, x)` is that whereas ...
- ... it correctly computes the subresultant prs,
- ... it computes the wrong signs of the Euclidean prs.
- Hence, `prem(f, g, x)` should be avoided.

Table of contents

Collins-Brown-Traub (CBT) subresultant prs algorithm 1/2

Collins-Brown-Traub (CBT) subresultant prs algorithm 1/2

- Since it is time consuming — and tiring — to evaluate a large number of determinants in order to compute the coefficients of the polynomials in a subresultant prs, the CBT subresultant prs algorithm computes the polynomials of the sequence as pseudo-remainders in $\mathbb{Z}[x]$ using the function `prem`.

Collins-Brown-Traub (CBT) subresultant prs algorithm 1/2

- Since it is time consuming — and tiring — to evaluate a large number of determinants in order to compute the coefficients of the polynomials in a subresultant prs, the CBT subresultant prs algorithm computes the polynomials of the sequence as pseudo-remainders in $\mathbb{Z}[x]$ using the function `prem`.

- If $LC(R_{i-1})$ is the leading coefficient of the divisor R_{i-1}, and $\delta = degree(R_{i-2}) - degree(R_{i-1}) + 1$, then (to exactly divide by the leading coefficient of the divisor) `prem` premultiplies the dividend times the leading coefficient of the divisor according to the formula

$$LC(R_{i-1})^{\delta} \cdot R_{i-2} = q_{i-2} \cdot R_{i-1} + R_i.$$

Collins-Brown-Traub (CBT) subresultant prs algorithm 2/2

Collins-Brown-Traub (CBT) subresultant prs algorithm 2/2

- This pre-multiplication, however, leads to an increase in the size of the coefficients, and a certain factor, β_i, is divided out to reduce the coefficients to subresultants.

Collins-Brown-Traub (CBT) subresultant prs algorithm 2/2

- This pre-multiplication, however, leads to an increase in the size of the coefficients, and a certain factor, β_i, is divided out to reduce the coefficients to subresultants.

- The expression β_i is given by the recurrence relations

$$\psi_1 = -1, \quad \beta_1 = (-1)^{\delta_1},$$
$$\psi_i = \frac{(-LC(R_{i-2}))^{\delta_{i-1}-1}}{\psi_{i-1}^{\delta_{i-1}-2}}, \qquad i > 1,$$
$$\beta_i = -LC(R_{i-2}) \cdot \psi_i^{\delta_i - 1}, \qquad i > 1,$$

where ...

Collins-Brown-Traub (CBT) subresultant prs algorithm 2/2

- This pre-multiplication, however, leads to an increase in the size of the coefficients, and a certain factor, β_i, is divided out to reduce the coefficients to subresultants.

- The expression β_i is given by the recurrence relations

$$\psi_1 = -1, \quad \beta_1 = (-1)^{\delta_1},$$
$$\psi_i = \frac{(-LC(R_{i-2}))^{\delta_{i-1}-1}}{\psi_{i-1}^{\delta_{i-1}-2}}, \qquad i > 1,$$
$$\beta_i = -LC(R_{i-2}) \cdot \psi_i^{\delta_i - 1}, \qquad i > 1,$$

where ...

- $$\delta_i = degree(R_{i-2}) - degree(R_{i-1}) + 1, \qquad i \geqslant 1.$$

Problem with the pseudo-remainder function `prem`

Problem with the pseudo-remainder function `prem`

- Given two polynomials in $\mathbb{Z}[x]$ consider the two sign sequences obtained from the Euclidean prs, computed with polynomial divisions performed in $\mathbb{Q}[x]$, and in $\mathbb{Z}[x]$ using `prem(f, g, x)`.

Problem with the pseudo-remainder function `prem`

- Given two polynomials in $\mathbb{Z}[x]$ consider the two sign sequences obtained from the Euclidean prs, computed with polynomial divisions performed in $\mathbb{Q}[x]$, and in $\mathbb{Z}[x]$ using `prem(f, g, x)`.

- If the a prs is complete, then the sign sequences are identical; that is, equation $LC(R_{i-1})^{\delta} \cdot R_{i-2} = q_{i-2} \cdot R_{i-1} + R_i$ can be safely used without any problem.

Problem with the pseudo-remainder function `prem`

- Given two polynomials in $\mathbb{Z}[x]$ consider the two sign sequences obtained from the Euclidean prs, computed with polynomial divisions performed in $\mathbb{Q}[x]$, and in $\mathbb{Z}[x]$ using `prem(f, g, x)`.

- If the a prs is complete, then the sign sequences are identical; that is, equation $LC(R_{i-1})^{\delta} \cdot R_{i-2} = q_{i-2} \cdot R_{i-1} + R_i$ can be safely used without any problem.

- However, if the prs is incomplete, then the sign sequences are not necessarily identical and this can be really confusing.

Problem with the pseudo-remainder function `prem`

- Given two polynomials in $\mathbb{Z}[x]$ consider the two sign sequences obtained from the Euclidean prs, computed with polynomial divisions performed in $\mathbb{Q}[x]$, and in $\mathbb{Z}[x]$ using `prem(f, g, x)`.

- If the a prs is complete, then the sign sequences are identical; that is, equation $LC(R_{i-1})^{\delta} \cdot R_{i-2} = q_{i-2} \cdot R_{i-1} + R_i$ can be safely used without any problem.

- However, if the prs is incomplete, then the sign sequences are not necessarily identical and this can be really confusing.

- After all, `prem(f, g, x)` is used to facilitate divisions in $\mathbb{Z}[x]$ and its purpose is not to distort sign sequences.

Example

Example

- Consider the polynomials $f = x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5$ and $g = 3x^6 + 5x^4 - 4x^2 - 9x + 21$ which result in an incomplete Euclidean prs of degrees 8, 6, 4, 2, 1, 0.

Example

- Consider the polynomials $f = x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5$ and $g = 3x^6 + 5x^4 - 4x^2 - 9x + 21$ which result in an incomplete Euclidean prs of degrees 8, 6, 4, 2, 1, 0.

- Performing polynomial divisions in $\mathbb{Q}[x]$ the sign sequence of the prs is

$$+, +, -, -, +, -$$

Example

- Consider the polynomials $f = x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5$ and $g = 3x^6 + 5x^4 - 4x^2 - 9x + 21$ which result in an incomplete Euclidean prs of degrees 8, 6, 4, 2, 1, 0.

- Performing polynomial divisions in $\mathbb{Q}[x]$ the sign sequence of the prs is

$$+, +, -, -, +, -$$

- On the other hand, performing polynomial divisions in $\mathbb{Z}[x]$ with the help of `prem(f, g, x)` the sign sequence of the prs is

$$+, +, -, +, +, +$$

Question

Question

- Would you use the function `prem(f, g, x)` on the polynomials of the previous example to compute their (generalized) Sturm sequence??

Question

- Would you use the function `prem(f, g, x)` on the polynomials of the previous example to compute their (generalized) Sturm sequence??

- Of course you would not ...

Question

- Would you use the function `prem(f, g, x)` on the polynomials of the previous example to compute their (generalized) Sturm sequence??

- Of course you would not ...

- ... even though it turns out that it does not disturb the sign sequence of this particular example.

Example where `prem(f, g, x)` fails in Sturm sequence

Example where `prem(f, g, x)` fails in Sturm sequence

- Consider the polynomials $f = 4x^5 - 3x^4 + 7$ and $g = 20x^4 - 12x^3$ which result in an incomplete Sturmian prs of degrees 5, 4, 3, 1, 0.

Example where `prem(f, g, x)` fails in Sturm sequence

- Consider the polynomials $f = 4x^5 - 3x^4 + 7$ and $g = 20x^4 - 12x^3$ which result in an incomplete Sturmian prs of degrees 5, 4, 3, 1, 0.

- Performing polynomial divisions in $\mathbb{Q}[x]$ the correct sign sequence of the Sturmian prs is

$$+, +, +, -, +$$

Example where `prem(f, g, x)` fails in Sturm sequence

- Consider the polynomials $f = 4x^5 - 3x^4 + 7$ and $g = 20x^4 - 12x^3$ which result in an incomplete Sturmian prs of degrees 5, 4, 3, 1, 0.

- Performing polynomial divisions in $\mathbb{Q}[x]$ the correct sign sequence of the Sturmian prs is

$$+, +, +, -, +$$

- On the other hand, performing polynomial divisions in $\mathbb{Z}[x]$ with the help of `prem(f, g, x)` the wrong sign sequence of the prs is

$$+, +, +, -, -$$

As a matter of fact the results obtained by the function `prem(f, g, x)` are wrong for incomplete sequences!

As a matter of fact the results obtained by the function `prem(f, g, x)` are wrong for incomplete sequences!

- The recently discovered theorem by Pell-Gordon (1917) allows us to correctly compute the signs of any Euclidean subresultant prs and renders `prem(f, g, x)` useless.

As a matter of fact the results obtained by the function `prem(f, g, x)` are wrong for incomplete sequences!

- The recently discovered theorem by Pell-Gordon (1917) allows us to correctly compute the signs of any Euclidean subresultant prs and renders `prem(f, g, x)` useless.

- We have written the `sympy` function `euclid_PG(f, g, x, method=0)` which performs polynomial divisions in $\mathbb{Q}[x]$ — and hence computes the correct signs; subsequently, the rationals are converted to integers using the Pell-Gordon theorem.

A more efficient solution using `remZ(f, g, x)`

A more efficient solution using `remZ(f, g, x)`

- Premultiply times the absolute value of the leading coefficient of the divisor; that is, use the function `remZ(f, g, x)`, which implements the equation

$$|LC(R_{i-1})|^{\delta} \cdot R_{i-2} = q_{i-2} \cdot R_{i-1} + R_i$$

A more efficient solution using `remZ(f, g, x)`

- Premultiply times the absolute value of the leading coefficient of the divisor; that is, use the function `remZ(f, g, x)`, which implements the equation

$$|LC(R_{i-1})|^{\delta} \cdot R_{i-2} = q_{i-2} \cdot R_{i-1} + R_i$$

- `remZ(f, g, x)` is equivalent to `euclid_PG(f, g, x, method=1)` and, hence, the sign sequences of Euclidean or Sturmian prs's in $\mathbb{Z}[x]$ are identical with the corresponding ones computed in $\mathbb{Q}[x]$.

A more efficient solution using `remZ(f, g, x)`

- Premultiply times the absolute value of the leading coefficient of the divisor; that is, use the function `remZ(f, g, x)`, which implements the equation

$$|LC(R_{i-1})|^{\delta} \cdot R_{i-2} = q_{i-2} \cdot R_{i-1} + R_i$$

- `remZ(f, g, x)` is equivalent to `euclid_PG(f, g, x, method=1)` and, hence, the sign sequences of Euclidean or Sturmian prs's in $\mathbb{Z}[x]$ are identical with the corresponding ones computed in $\mathbb{Q}[x]$.

- Collateral damage: Using `remZ(f, g, x)` the CBT subresultant prs algorithm does not work properly. Hence, new methods need to be developed!

Table of contents

Preliminaries

Preliminaries

- All methods presented in this talk were inspired by the Pell-Gordon theorem of 1917 — discovered by P. S. Vigklas. So we are going to begin with it. But keep in mind ...

Preliminaries

- All methods presented in this talk were inspired by the Pell-Gordon theorem of 1917 — discovered by P. S. Vigklas. So we are going to begin with it. But keep in mind ...

- ... that Pell and Gordon perform the polynomial divisions in $\mathbb{Q}[x]$ and that ....

Preliminaries

- All methods presented in this talk were inspired by the Pell-Gordon theorem of 1917 — discovered by P. S. Vigklas. So we are going to begin with it. But keep in mind ...

- ... that Pell and Gordon perform the polynomial divisions in $\mathbb{Q}[x]$ and that ....

- ... they evaluate determinants of submatrices of `sylvester2`.

Preliminaries

- All methods presented in this talk were inspired by the Pell-Gordon theorem of 1917 — discovered by P. S. Vigklas. So we are going to begin with it. But keep in mind ...

- ... that Pell and Gordon perform the polynomial divisions in $\mathbb{Q}[x]$ and that ....

- ... they evaluate determinants of submatrices of `sylvester2`.

- It is worth noting that the `sylvester2` matrix is equivalent to the `bezout` matrix.

The Pell-Gordon theorem 1/3

The Pell-Gordon theorem 1/3

Theorem

(Pell-Gordon, 1917) *Let* $A = a_0x^n + a_1x^{n-1} + \cdots + a_n$ *and* $B = b_0x^n + b_1x^{n-1} + \cdots + b_n$ *be two polynomials of the n-th degree. Modify the process of finding the highest common factor of A and B by taking at each stage the negative of the remainder. Let the i-th modified remainder be*

$$R^{(i)} = r_0^{(i)}x^{m_i} + r_1^{(i)}x^{m_i-1} + \cdots + r_{m_i}^{(i)}$$

where $(m_i + 1)$ *is the degree of the preceding remainder, and where the first* $(p_i - 1)$ *coefficients of* $R^{(i)}$ *are zero, and the* p_i*-th coefficient* $\varrho_i = r_{p_i-1}^{(i)}$ *is different from zero. Then ...*

The Pell-Gordon theorem 2/3

The Pell-Gordon theorem 2/3

Theorem

... *for* $k = 0, 1, \ldots, m_i$ *the coefficients* $r_k^{(i)}$ *are given by*

$$r_k^{(i)} = \frac{(-1)^{u_{i-1}}(-1)^{u_{i-2}} \cdots (-1)^{u_1}(-1)^{v_{i-1}}}{\varrho_{i-1}^{p_{i-1}+1}\varrho_{i-2}^{p_{i-2}+p_{i-1}} \cdots \varrho_1^{p_1+p_2}\varrho_0^{p_1}} \cdot \det(i, k),$$

where $u_{i-1} = 1 + 2 + \cdots + p_{i-1}$, $\quad v_{i-1} = p_1 + p_2 + \cdots + p_{i-1}$ *and ...*

The Pell-Gordon theorem 3/3

The Pell-Gordon theorem 3/3

Theorem

...

$$\det(i,k) = \begin{vmatrix} a_0 & a_1 & a_2 & \cdots & \cdot & \cdot & \cdots & a_{2v_{i-1}} & a_{2v_{i-1}+1+k} \\ b_0 & b_1 & b_2 & \cdots & \cdot & \cdot & \cdots & b_{2v_{i-1}} & b_{2v_{i-1}+1+k} \\ 0 & a_0 & a_1 & \cdots & \cdot & \cdot & \cdots & a_{2v_{i-1}-1} & a_{2v_{i-1}+k} \\ 0 & b_0 & b_1 & \cdots & \cdot & \cdot & \cdots & b_{2v_{i-1}-1} & b_{2v_{i-1}+k} \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & a_0 & a_1 & \cdots & a_{v_{i-1}} & a_{v_{i-1}+1+k} \\ 0 & 0 & 0 & \cdots & b_0 & b_1 & \cdots & b_{v_{i-1}} & b_{v_{i-1}+1+k} \end{vmatrix}.$$

Remarks 1/2

Remarks 1/2

- It is understood in that $\varrho_0 = b_0$, $p_0 = 0$, and that $a_i = b_i = 0$ for $i > n$.

Remarks 1/2

- It is understood in that $\varrho_0 = b_0$, $p_0 = 0$, and that $a_i = b_i = 0$ for $i > n$.

- We modified the formula as shown below with the changes appearing in red

$$r_k^{(i)} = \frac{(-1)^{u_{i-1}} (-1)^{u_{i-2}} \cdots (-1)^{u_1} (-1)^{u_0} (-1)^{v_{i-1}}}{\varrho_{i-1}^{p_{i-1}+\mathbf{p_i}-\mathbf{degDiffer}} \varrho_{i-2}^{p_{i-2}+p_{i-1}} \cdots \varrho_1^{p_1+p_2} \varrho_0^{\mathbf{p_0}+p_1}} \cdot \frac{\det(i,k)}{\varrho_{\mathbf{-1}}},$$

where ...

Remarks 1/2

- It is understood in that $\varrho_0 = b_0$, $p_0 = 0$, and that $a_i = b_i = 0$ for $i > n$.

- We modified the formula as shown below with the changes appearing in red

$$r_k^{(i)} = \frac{(-1)^{u_{i-1}} (-1)^{u_{i-2}} \cdots (-1)^{u_1} (-1)^{u_0} (-1)^{v_{i-1}}}{\varrho_{i-1}^{p_{i-1}+\mathbf{p_i}-\mathbf{degDiffer}} \varrho_{i-2}^{p_{i-2}+p_{i-1}} \cdots \varrho_1^{p_1+p_2} \varrho_0^{\mathbf{p_0}+p_1}} \cdot \frac{\det(i,k)}{\varrho_{\mathbf{-1}}},$$

where ...

- ... $\varrho_{-1} = a_0$, is the leading coefficient of A and `degDiffer` is the difference between the expected degree m_i and the actual degree of the remainder.

Remarks 2/2

Remarks 2/2

- Clearly, using the modified formula mentioned in the previous frame and multiplying times the absolute value of the denominator we uniquely determine the complete or incomplete Sturmian polynomial remainders in $\mathbb{Z}[x]$.

The main application of the Pell-Gordon theorem

The main application of the Pell-Gordon theorem

- In 1900 Van Vleck developed a method to compute the complete Sturm sequence (prs) of a polynomial f by triangularizing the `sylvester2` matrix of f, f'.

The main application of the Pell-Gordon theorem

- In 1900 Van Vleck developed a method to compute the complete Sturm sequence (prs) of a polynomial f by triangularizing the `sylvester2` matrix of f, f'.

- In 1986 Akritas "generalized" Van Vleck's triangularization method to both complete and incomplete subresultant prs's but for the latter case he could not get the signs right.

The main application of the Pell-Gordon theorem

- In 1900 Van Vleck developed a method to compute the complete Sturm sequence (prs) of a polynomial f by triangularizing the `sylvester2` matrix of f, f'.

- In 1986 Akritas "generalized" Van Vleck's triangularization method to both complete and incomplete subresultant prs's but for the latter case he could not get the signs right.

- The failed attempts to solve the "signs problem" lasted close to 30 years, until P.S. Vigklas discovered the Pell-Gordon theorem.

The main application of the Pell-Gordon theorem

- In 1900 Van Vleck developed a method to compute the complete Sturm sequence (prs) of a polynomial f by triangularizing the `sylvester2` matrix of f, f'.

- In 1986 Akritas "generalized" Van Vleck's triangularization method to both complete and incomplete subresultant prs's but for the latter case he could not get the signs right.

- The failed attempts to solve the "signs problem" lasted close to 30 years, until P.S. Vigklas discovered the Pell-Gordon theorem.

- Pell and Gordon showed the relationship that exists between the polynomials of an incomplete prs computed by divisions in $\mathbb{Q}[x]$ and the ones whose coefficients are computed as modified subresultants (determinants of submatrices) of the `sylvester2` matrix.

The essence of the Pell-Gordon theorem 1/2

The essence of the Pell-Gordon theorem 1/2

▶ The Pell-Gordon theorem proved that if we want to compute an incomplete Sturm sequence by Van Vleck's matrix triangularization method, the only way to obtain the correct signs is to evaluate — for each polynomial remainder — one modified subresultant (determinant of a submatrix) of the `sylvester2` matrix.

The essence of the Pell-Gordon theorem 1/2

▶ The Pell-Gordon theorem proved that if we want to compute an incomplete Sturm sequence by Van Vleck's matrix triangularization method, the only way to obtain the correct signs is to evaluate — for each polynomial remainder — one modified subresultant (determinant of a submatrix) of the `sylvester2` matrix.

▶ In other words, the essence — or the main idea — of the Pell-Gordon theorem is that we can now use determinant evaluation as a new method for computing modified subresultant prs's.

The essence of the Pell-Gordon theorem prs's 2/2

The essence of the Pell-Gordon theorem prs's 2/2

- We used the above idea not only to compute the correct signs but also to force all coefficients of the polynomials in the Sturm sequence to become modified subresultants of the `sylvester2` matrix. This way, Van Vleck's triangularization method works also for incomplete prs's and, we are thus able to compute modified subresultant prs's of any kind.

The essence of the Pell-Gordon theorem prs's 2/2

- We used the above idea not only to compute the correct signs but also to force all coefficients of the polynomials in the Sturm sequence to become modified subresultants of the `sylvester2` matrix. This way, Van Vleck's triangularization method works also for incomplete prs's and, we are thus able to compute modified subresultant prs's of any kind.

- Variations of this main idea have been used to develop new methods for computing (proper) subresultant prs's of any kind. These methods are discussed below.

Table of contents

The main idea

The main idea

- As mentioned before, we decided to use the essential idea of the Pell-Gordon theorem, in order to compute subresultant prs's...

The main idea

- As mentioned before, we decided to use the essential idea of the Pell-Gordon theorem, in order to compute subresultant prs's...

- ... where now — for each polynomial remainder — we evaluate one subresultant (determinant of a submatrix) of the `sylvester1` matrix.

The main idea

- As mentioned before, we decided to use the essential idea of the Pell-Gordon theorem, in order to compute subresultant prs's...

- ... where now — for each polynomial remainder — we evaluate one subresultant (determinant of a submatrix) of the `sylvester1` matrix.

- This technique is applied in all three new methods for computing subresultant prs's. Moreover, instead of `sylvester1` we can also use the `bezout` matrix, which has smaller dimensions.

Details of the main idea 1/2

Details of the main idea 1/2

- Suppose that we have computed the i-th polynomial remainder of *expected* degree (expDeg) m_i

$$s^{(i)} = s_0^{(i)} x^{m_i} + s_1^{(i)} x^{m_i - 1} + \cdots + s_{m_i}^{(i)},$$

where all the coefficients $s_k^{(i)}, 0 \leq k \leq m_i$ are either integers or rationals, depending on the method used.

Details of the main idea 1/2

- Suppose that we have computed the i-th polynomial remainder of *expected* degree (`expDeg`) m_i

$$s^{(i)} = s_0^{(i)} x^{m_i} + s_1^{(i)} x^{m_i - 1} + \cdots + s_{m_i}^{(i)},$$

where all the coefficients $s_k^{(i)}, 0 \leq k \leq m_i$ are either integers or rationals, depending on the method used.

- To compute the correct sign of the polynomial $s^{(i)}$ and to make its coefficients subresultants we evaluate the determinant $\det(i,j) \neq 0$ of the sub-matrix of `sylvester1` corresponding to the *actual* leading coefficient $s_j^{(i)}$ of $s^{(i)}$.

Details of the main idea 2/2

Details of the main idea 2/2

- Then by forming the product

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i, j)$$

we obtain the i-th polynomial of the subresultant prs.

Details of the main idea 2/2

- Then by forming the product

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

we obtain the i-th polynomial of the subresultant prs.

- This is so, because of the existing ratio equality

$$\left| \frac{\det(i,j)}{s_j^{(i)}} \right| = \left| \frac{\det(i,k)}{s_k^{(i)}} \right|$$

that holds for $j < k \leq m_i$.

Details of the main idea 2/2

- Then by forming the product

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

we obtain the i-th polynomial of the subresultant prs.

- This is so, because of the existing ratio equality

$$\left| \frac{\det(i,j)}{s_j^{(i)}} \right| = \left| \frac{\det(i,k)}{s_k^{(i)}} \right|$$

that holds for $j < k \leq m_i$.

- Recall that $s_j^{(i)}$ is the *actual* leading coefficient of the polynomial remainder $s^{(i)}$.

Outline of the first method

Outline of the first method

- This new method performs operations in $\mathbb{Z}[x]$ and computes the subresultant prs by:

Outline of the first method

▶ This new method performs operations in $\mathbb{Z}[x]$ and computes the subresultant prs by:

- premultipling the dividend times the *absolute* value of the leading coefficient of the divisor (`remZ`),

Outline of the first method

▶ This new method performs operations in $\mathbb{Z}[x]$ and computes the subresultant prs by:

- premultipling the dividend times the *absolute* value of the leading coefficient of the divisor (`remZ`),

- evaluating the subresultant of `sylvester1` corresponding to the leading coefficient of the remainder in order to obtain its correct sign and to force its coefficients to become subresultants *without* the quantities β_i of the CBT method.

In other words ...

In other words ...

- If $s^{(i)}$ is the i-th remainder computed with `remZ`, $s_j^{(i)}$, is its leading coefficient and $\det(i, j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

In other words ...

- If $s^{(i)}$ is the i-th remainder computed with `remZ`, $s_j^{(i)}$, is its leading coefficient and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

- ... then

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

is the i-th polynomial of the subresultant prs.

In other words ...

- If $s^{(i)}$ is the i-th remainder computed with `remZ`, $s_j^{(i)}$, is its leading coefficient and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

- ... then

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

is the i-th polynomial of the subresultant prs.

- Clearly, in the above formula, the sign and the value of the leading coefficient of the resulting polynomial are those of the determinant.

Advantage of the method

Advantage of the method

- The strong and most important point of this new method is that the function `remZ` does *not* distort the sign sequence of the Euclidean prs computed either in $\mathbb{Q}[x]$ or in $\mathbb{Z}[x]$ with `remZ`.

Advantage of the method

- The strong and most important point of this new method is that the function `remZ` does *not* distort the sign sequence of the Euclidean prs computed either in $\mathbb{Q}[x]$ or in $\mathbb{Z}[x]$ with `remZ`.

- We have implemented this method in sympy as the function `subresultants_remZ(f, g, x)`, and have made it available for use.

Outline of the second method — original version

Outline of the second method — original version

- With this method we compute each remainder by performing polynomial divisions in $\mathbb{Q}[x]$.

Outline of the second method — original version

- With this method we compute each remainder by performing polynomial divisions in $\mathbb{Q}[x]$.

- To force the coefficients of each remainder into $\mathbb{Z}[x]$ we multiply the remainder times the absolute value of the denominator in the (modified) formula of the Pell-Gordon Theorem (note that $\det(i,k)$ is a modified subresultant of `sylvester2`)

$$r_k^{(i)} = \frac{(-1)^{u_{i-1}}(-1)^{u_{i-2}}\cdots(-1)^{u_1}(-1)^{u_0}(-1)^{v_{i-1}}}{\varrho_{i-1}^{p_{i-1}+\mathbf{p_i}-\mathbf{degDiffer}}\varrho_{i-2}^{p_{i-2}+p_{i-1}}\cdots\varrho_1^{p_1+p_2}\varrho_0^{\mathbf{p_0}+p_1}}\cdot\frac{\det(i,k)}{\varrho_{-\mathbf{1}}}.$$

Outline of the second method — original version

- With this method we compute each remainder by performing polynomial divisions in $\mathbb{Q}[x]$.

- To force the coefficients of each remainder into $\mathbb{Z}[x]$ we multiply the remainder times the absolute value of the denominator in the (modified) formula of the Pell-Gordon Theorem (note that $\det(i,k)$ is a modified subresultant of `sylvester2`)

$$r_k^{(i)} = \frac{(-1)^{u_{i-1}}(-1)^{u_{i-2}}\cdots(-1)^{u_1}(-1)^{u_0}(-1)^{v_{i-1}}}{\varrho_{i-1}^{p_{i-1}+\mathbf{p_i}-\mathbf{degDiffer}}\varrho_{i-2}^{p_{i-2}+p_{i-1}}\cdots\varrho_1^{p_1+p_2}\varrho_0^{\mathbf{p_0}+p_1}}\cdot\frac{\det(i,k)}{\varrho_{-\mathbf{1}}}.$$

- Then, the correct sign of each remainder is obtained by evaluating the sign of the subresultant (of `sylvester1` or `bezout`) corresponding to its leading coefficient.

Implementation

Implementation

- The above procedure is easily programmed. The only critical point is to effectively compute the absolute value of the denominator in the modified formula.

Implementation

- The above procedure is easily programmed. The only critical point is to effectively compute the absolute value of the denominator in the modified formula.

- This value is not computed anew for each remainder $R^{(i)}$; instead, a multiplication factor, `mulFac`, is being updated as new leading coefficients are included in the expression.

Implementation

- The above procedure is easily programmed. The only critical point is to effectively compute the absolute value of the denominator in the modified formula.

- This value is not computed anew for each remainder $R^{(i)}$; instead, a multiplication factor, `mulFac`, is being updated as new leading coefficients are included in the expression.

- We have implemented this method in sympy as the function `subresultants_PG(f, g, x)`, and have made it available for use.

Outline of the second method — a more efficient version

Outline of the second method — a more efficient version

- If $s^{(i)}$ is the i-th polynomial remainder computed with polynomial division in $\mathbb{Q}[x]$, $s_j^{(i)}$ is its leading coefficient, and $\det(i, j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

Outline of the second method — a more efficient version

- If $s^{(i)}$ is the i-th polynomial remainder computed with polynomial division in $\mathbb{Q}[x]$, $s_j^{(i)}$ is its leading coefficient, and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

- ... then

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

is the i-th polynomial of the subresultant prs.

Outline of the second method — a more efficient version

- If $s^{(i)}$ is the i-th polynomial remainder computed with polynomial division in $\mathbb{Q}[x]$, $s_j^{(i)}$ is its leading coefficient, and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

- ... then

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

is the i-th polynomial of the subresultant prs.

- We have implemented this method in sympy as the function `subresultants_PG2(f, g, x)`, and have made it available for use.

It is more efficient because ...

It is more efficient because ...

- We now use only `sylvester1` to both convert the rational coefficients of each remainder to absolute values of subresultants and simultaneously compute the correct sign of its leading coefficient.

It is more efficient because ...

- We now use only `sylvester1` to both convert the rational coefficients of each remainder to absolute values of subresultants and simultaneously compute the correct sign of its leading coefficient.

- In other words, we do not have to keep track of the absolute value of the denominator in the modified formula.

Remarks on the third method

Remarks on the third method

- The third method is a slight variation of the Van Vleck-Pell-Gordon procedure for computing Sturm sequences (modified subresultant prs's) by triangularizing the `sylvester2` matrix.

Remarks on the third method

- The third method is a slight variation of the Van Vleck-Pell-Gordon procedure for computing Sturm sequences (modified subresultant prs's) by triangularizing the `sylvester2` matrix.

- It is worth noting that with this method, and for the first time in the literature, we have an explicit interplay between the two Sylvester matrices, `sylvester1` and `sylvester2`.

Outline of the third method — original version 1/2

Outline of the third method — original version 1/2

- With this method we do not perform polynomial divisions; instead, the remainders are computed in $\mathbb{Z}[x]$ by matrix triangularization.

Outline of the third method — original version 1/2

- With this method we do not perform polynomial divisions; instead, the remainders are computed in $\mathbb{Z}[x]$ by matrix triangularization.

- We triangularize Sylvester's matrix `sylvester2` to compute each polynomial - candidate for the subresultant prs and, then, we evaluate the subresultant (determinant of a submatrix) of `sylvester1`, corresponding to the leading coefficient of the candidate under consideration, to make it a member of the subresultant prs.

Outline of the third method — original version 2/2

Outline of the third method — original version 2/2

- In other words, if $s^{(i)}$ is the i-th polynomial remainder computed in $\mathbb{Z}[x]$ by triangularizing `sylvester2`, $s_j^{(i)}$ is its leading coefficient, and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

Outline of the third method — original version 2/2

- In other words, if $s^{(i)}$ is the i-th polynomial remainder computed in $\mathbb{Z}[x]$ by triangularizing `sylvester2`, $s_j^{(i)}$ is its leading coefficient, and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

- ... then

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

is the i-th polynomial of the subresultant prs.

Outline of the third method — original version 2/2

- In other words, if $s^{(i)}$ is the i-th polynomial remainder computed in $\mathbb{Z}[x]$ by triangularizing `sylvester2`, $s_j^{(i)}$ is its leading coefficient, and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

- ... then

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

is the i-th polynomial of the subresultant prs.

- We have implemented this method in `sympy` as the function `subresultants_triang(f, g, x, method=0)`. Set `method=1` to print out the triangularized matrix.

Outline of the third method — a more efficient version 1/2

Outline of the third method — a more efficient version 1/2

- In the more efficient method we take advantage of the fact that polynomial division is equivalent to the triangularization of a "small" matrix. Hence, we do not triangularize the `sylvester2` matrix, but a number of smaller matrices from each one of which we obtain one polynomial - candidate for the subresultant prs.

Outline of the third method — a more efficient version 1/2

- In the more efficient method we take advantage of the fact that polynomial division is equivalent to the triangularization of a "small" matrix. Hence, we do not triangularize the `sylvester2` matrix, but a number of smaller matrices from each one of which we obtain one polynomial - candidate for the subresultant prs.

- We then evaluate the subresultant (determinant of the submatrix) of `sylvester1` corresponding to the leading coefficient of each polynomial - candidate, and turn it into a member of the subresultant prs by adjusting its sign and coefficients.

Outline of the second method — a more efficient version 2/2

Outline of the second method — a more efficient version 2/2

- In other words, if $s^{(i)}$ is the i-th polynomial remainder computed in $\mathbb{Z}[x]$ by triangularizing a "smaller" matrix, $s_j^{(i)}$ is its leading coefficient, and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

Outline of the second method — a more efficient version 2/2

- In other words, if $s^{(i)}$ is the i-th polynomial remainder computed in $\mathbb{Z}[x]$ by triangularizing a "smaller" matrix, $s_j^{(i)}$ is its leading coefficient, and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

- ... then

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

is the i-th polynomial of the subresultant prs.

Outline of the second method — a more efficient version 2/2

- In other words, if $s^{(i)}$ is the i-th polynomial remainder computed in $\mathbb{Z}[x]$ by triangularizing a "smaller" matrix, $s_j^{(i)}$ is its leading coefficient, and $\det(i,j)$ is the subresultant of `sylvester1` corresponding to $s_j^{(i)}$, ...

- ... then

$$\frac{s^{(i)}}{s_j^{(i)}} \cdot \det(i,j)$$

is the i-th polynomial of the subresultant prs.

- We have implemented this method in sympy as the function `subresultants_triang2(f, g, x)`, and have made it available for use.

Table of contents

References

References

- Akritas, A.G., Malaschonok, G.I., Vigklas, P.S.: "On a Theorem by Van Vleck Regarding Sturm Sequences." Serdica Journal of Computing, Vol. 7, No 4, 101–134, 2013.
- Akritas, A.G., Malaschonok, G.I., Vigklas, P.S.: "Sturm Sequences and Modified Subresultant Polynomial Remainder Sequences". Serdica Journal of Computing, to appear.
- Akritas, A. G.: "Three New Methods for Computing Subresultant Polynomial Remainder Sequences (PRS's)". Serdica Journal of Computing, to appear.