Popov Forms of Matrices of Differential Polynomials

George Labahn

Symbolic Computation Group
Cheriton School of Computer Science
University of Waterloo, Canada

May 13, 2009

Example

Consider system of differential equations

$$\begin{array}{rcrcrcl} y_1''(t) + (t+2)y_1(t) & + & t^2 y_2''(t) + y_2(t) & + & y_3'(t) + y_3(t) & = & 0 \\ y_1'(t) + 3y_1(t) & + & y_2'''(t) + 2y_2'(t) - y_2(t) & + & y_3'''(t) - 2t^2 y_3(t) & = & 0 \\ y_1'(t) + y_1(t) & + & y_2''(t) + 2t y_2'(t) - y_2(t) & + & y_3''''(t) & = & 0. \end{array}$$

We usually deal with such systems by first converting them to first order systems

$$A(t)Y'(t) = B(t)Y(t) + C(t)$$

and then using various techniques to build various solutions or solution types (e.g. existence of rational function or exponential solutions).

Example : Matrix Form

Our original example can be represented by a differential matrix equation

$$\begin{bmatrix} D^2+(t+2) & t^2D^2+1 & D+1 \\ D+3 & D^3+2D-1 & D^3-2t^2 \\ D+1 & D^2+2tD+1 & D^4 \end{bmatrix} \cdot \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{bmatrix} = \mathbf{0}.$$

In general, systems that we are looking at are of the form

$$A(D)Y(t) = B(t).$$

Question : What form does $A(D)$ need to be in order that one can convert easily to a first order system?

Example : Matrix Form

Our original example can be represented by a differential matrix equation

$$\begin{bmatrix} D^2 + (t+2) & t^2 D^2 + 1 & D + 1 \\ D + 3 & D^3 + 2D - 1 & D^3 - 2t^2 \\ D + 1 & D^2 + 2tD + 1 & D^4 \end{bmatrix} \cdot \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{bmatrix} = \mathbf{0}.$$

In general, systems that we are looking at are of the form

$$A(D)Y(t) = B(t).$$

Question : What form does $A(D)$ need to be in order that one can convert easily to a first order system?

Example (cont.)

Let D be the differentiation operator on t. If the system of equations is represented by:

$$\begin{bmatrix} D^2 + (t+2) & t^2D^2 + 1 & D + 1 \\ D + 3 & D^3 + 2D - 1 & D^3 - 2t^2 \\ D + 1 & D^2 + 2tD + 1 & D^4 \end{bmatrix} \cdot \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{bmatrix} = \mathbf{0},$$

then we can rewrite

$$\begin{aligned} y_1''(t) &= -(t+2)y_1(t) - t^2y_2''(t) - y_2(t) - y_3'(t) - y_3(t) \\ y_2'''(t) &= -y_1'(t) - 3y_1(t) - 2y_2'(t) + y_2(t) - y_3'''(t) + 2t^2y_3(t) \\ y_3''''(t) &= -y_1'(t) - y_1(t) - y_2''(t) - 2ty_2'(t) - y_2(t) \end{aligned}$$

Example (cont.)

- For systems not having this ‘special form’ one can always do row operations, derivations and eliminations to put a matrix of differential operators into the correct form.
- Basically given $A(D)$ one looks for an invertible $U(D)$ such that

$$U(D) \cdot A(D) = P(D) = \text{ matrix in special form}$$

- Special form needs to have columns of highest order in each row and one row cannot ‘interfere’ with columns of higest order in other rows.

Questions

- What are these special normal forms?
- How to compute such normal forms?
- Where does one go for ideas for these normal forms?

WARNING : this is only a preliminary report on this topic.

Outline

Outline

Todays Topic

Given : $\mathbf{A}(D) \in \mathbb{K}^{m\times n}[D]$.

Do row operations

$$\mathbf{A}(D) = \text{ easier}$$

Todays Topic

Given : $\mathbf{A}(D) \in \mathbb{K}^{m\times n}[D]$.

Do row operations $\mathbf{U}(D)$

$$\mathbf{U}(D)\mathbf{A}(D) = \text{ easier}$$

(easier = $\mathbf{B}(D) \in \mathbb{K}^{m\times n}[D]$ in some sort of normal form)

Todays Topic

Given : $\mathbf{A}(D) \in \mathbb{K}^{m\times n}[D]$.

Do row operations $\mathbf{U}(D)$

$$\mathbf{U}(D)\mathbf{A}(D) = \text{ easier}$$

(easier = $\mathbf{B}(D) \in \mathbb{K}^{m\times n}[D]$ in some sort of normal form)

$\mathbf{U}(D) \in \mathbb{K}^{m\times m}[D]$ invertible

Todays Topic

Given : $\mathbf{A}(D) \in \mathbb{K}^{m\times n}[D]$.

Do row operations $\mathbf{U}(D)$

$$\mathbf{U}(D)\mathbf{A}(D) = \text{ easier}$$

(easier = $\mathbf{B}(D) \in \mathbb{K}^{m\times n}[D]$ in some sort of normal form)

$\mathbf{U}(D) \in \mathbb{K}^{m\times m}[D]$ invertible

Also wish to do this with matrices of Ore operators

Useful to see how one does these with matrices of polynomials

Why useful for Matrix Polynomials? : Matrix GCD

Given $B(z), C(z) \in \mathbb{K}^{m \times m}[z]$:

Find **Greatest Right Common Divisor** (gcrd) $D(z) \in \mathbb{K}^{m \times m}[z]$.

Why useful for Matrix Polynomials? : Matrix GCD

Given $B(z), C(z) \in \mathbb{K}^{m\times m}[z]$:

Find **Greatest Right Common Divisor** (gcrd) $D(z) \in \mathbb{K}^{m\times m}[z]$.

$$\begin{bmatrix} B(z) \\ C(z) \end{bmatrix}$$

Why useful for Matrix Polynomials? : Matrix GCD

Given $B(z), C(z) \in \mathbb{K}^{m\times m}[z]$:

Find **Greatest Right Common Divisor** (gcrd) $D(z) \in \mathbb{K}^{m\times m}[z]$.

$$\begin{bmatrix} U_{11}(z) & U_{12}(z) \\ U_{21}(z) & U_{22}(z) \end{bmatrix} \cdot \begin{bmatrix} B(z) \\ C(z) \end{bmatrix} = \begin{bmatrix} D(z) \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} V_{11}(z) & V_{12}(z) \\ V_{21}(z) & V_{22}(z) \end{bmatrix} \cdot \begin{bmatrix} U_{11}(z) & U_{12}(z) \\ U_{21}(z) & U_{22}(z) \end{bmatrix} = \begin{bmatrix} I_m & 0 \\ 0 & I_m \end{bmatrix}$$

Why useful for Matrix Polynomials? : Matrix GCD

Given $B(z), C(z) \in \mathbb{K}^{m\times m}[z]$:

Find **Greatest Right Common Divisor** (gcrd) $D(z) \in \mathbb{K}^{m\times m}[z]$.

$$\begin{bmatrix} B(z) \\ C(z) \end{bmatrix} = \begin{bmatrix} V_{11}(z)D(z) \\ V_{21}(z)D(z) \end{bmatrix}$$

$$\begin{bmatrix} U_{11}(z) & U_{12}(z) \\ U_{21}(z) & U_{22}(z) \end{bmatrix} \cdot \begin{bmatrix} V_{11}(z) & V_{12}(z) \\ V_{21}(z) & V_{22}(z) \end{bmatrix} = \begin{bmatrix} I_m & 0 \\ 0 & I_m \end{bmatrix}$$

Why useful for Matrix Polynomials? : Matrix GCD

Given $B(z), C(z) \in \mathbb{K}^{m\times m}[z]$:

Find **Greatest Right Common Divisor** (gcrd) $D(z) \in \mathbb{K}^{m\times m}[z]$.

$$\begin{bmatrix} B(z) \\ C(z) \end{bmatrix} = \begin{bmatrix} V_{11}(z)D(z) \\ V_{21}(z)D(z) \end{bmatrix}$$

$$U_{11}(z)V_{11}(z) + U_{12}(z)V_{21}(z) = I_m$$

- Matrix polynomials (in fact rational expressions of form $A(z) = U(z) \cdot V(z)^{-1}$) used in linear control theory

- Matrix GCDs needed for minimal rational matrix expressions

- Builds input-output model for control system

- Concept of Transfer functions also seems to exist for nonlinear control (Ziming Li [FoCM'08])

Outline

Example : Hermite Normal Form

$$\mathbf{H}(z) = \begin{bmatrix} h_{1,1}(z) & h_{1,2}(z) & \cdots & h_{1,m}(z) \\ 0 & h_{2,2}(z) & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & & h_{m-1,m}(z) \\ 0 & \cdots & 0 & h_{m,m}(z) \end{bmatrix}$$

is in Hermite Normal Form if:

- Upper triangular

Example : Hermite Normal Form

$$\mathbf{H}(z) = \begin{bmatrix} h_{1,1}(z) & h_{1,2}(z) & \cdots & h_{1,m}(z) \\ 0 & h_{2,2}(z) & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & & h_{m-1,m}(z) \\ 0 & \cdots & 0 & h_{m,m}(z) \end{bmatrix}$$

is in Hermite Normal Form if:

- Upper triangular
- diagonal entries monic

Example : Hermite Normal Form

$$\mathbf{H}(z) = \begin{bmatrix} h_{1,1}(z) & h_{1,2}(z) & \cdots & h_{1,m}(z) \\ 0 & h_{2,2}(z) & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & & h_{m-1,m}(z) \\ 0 & \cdots & 0 & h_{m,m}(z) \end{bmatrix}$$

is in Hermite Normal Form if:

- Upper triangular
- diagonal entries monic
- degrees of diagonal entries max in columns

Example : Hermite Normal Form

$$\mathbf{H}(z) = \begin{bmatrix} h_{1,1}(z) & h_{1,2}(z) & \cdots & h_{1,m}(z) \\ 0 & h_{2,2}(z) & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & & h_{m-1,m}(z) \\ 0 & \cdots & 0 & h_{m,m}(z) \end{bmatrix}$$

is in Hermite Normal Form if:

- Upper triangular
- diagonal entries monic
- degrees of diagonal entries max in columns
- any zero rows at bottom

Example : Hermite Normal Form

$$\mathbf{H}(z) = \begin{bmatrix} h_{1,1}(z) & h_{1,2}(z) & \cdots & h_{1,m}(z) \\ 0 & h_{2,2}(z) & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & & h_{m-1,m}(z) \\ 0 & \cdots & 0 & h_{m,m}(z) \end{bmatrix}$$

is in Hermite Normal Form if:

- Upper triangular
- diagonal entries monic
- degrees of diagonal entries max in columns
- any zero rows at bottom

Useful in solving linear system $\mathbf{H}(z)\text{vec}x(z) = \text{vec}b(z)$

Example

$$\text{Input} : A(z) = \begin{bmatrix} z^2+1 & z & z^3 \\ z & 0 & z \\ z & z & z^3-1 \end{bmatrix}$$

$$\text{Output} : B(z) = \begin{bmatrix} 1 & 0 & -z^2+z+1 \\ 0 & z & z^2-z-1 \\ 0 & 0 & z^3-z^2 \end{bmatrix}$$

Some Additional Remarks

- Also have **Smith Normal Form** for row and column equivalence.

$$\mathbf{U}(z) \cdot \mathbf{A}(z) \cdot \mathbf{V}(z) = diag(s_1(z), \cdots, s_m(z))$$

where $s_i(z) | s_{i+1}(z)$ for all i. Determinantal divisors. Invariant factors. Useful for solving

$$\mathbf{A}(z)\text{vec}x(z) = \text{vec}b(z).$$

- Also have noncommutative versions of these normal forms
  - e.g. for matrices $\mathbf{A}(D)$ of differential operators
  - again useful for solving systems, but now of the form

$$\mathbf{A}(D)\text{vec}x(z) = \text{vec}b(z).$$

  - e.g. used by Singer [1985] for LODE decision procedures for systems

This talk : Popov Form

- Hermite Normal Form does not have controlled degrees

This talk : Popov Form

- Hermite Normal Form does not have controlled degrees
  - e.g. degrees of HNF can be larger than input degree

This talk : Popov Form

- Hermite Normal Form does not have controlled degrees
  - e.g. degrees of HNF can be larger than input degree
- Popov's form (1969) : purpose was to allow for simple conversion of state space to transfer functions in linear systems theory.

This talk : Popov Form

- Hermite Normal Form does not have controlled degrees
  - e.g. degrees of HNF can be larger than input degree
- Popov's form (1969) : purpose was to allow for simple conversion of state space to transfer functions in linear systems theory.
- Villard (1996) introduced Popov form to computer algebra community

This talk : Popov Form

- Hermite Normal Form does not have controlled degrees
  - e.g. degrees of HNF can be larger than input degree
- Popov's form (1969) : purpose was to allow for simple conversion of state space to transfer functions in linear systems theory.
- Villard (1996) introduced Popov form to computer algebra community
- Popov form related to Gröbner bases

This talk : Popov Form

- Hermite Normal Form does not have controlled degrees
  - e.g. degrees of HNF can be larger than input degree
- Popov's form (1969) : purpose was to allow for simple conversion of state space to transfer functions in linear systems theory.
- Villard (1996) introduced Popov form to computer algebra community
- Popov form related to Gröbner bases
- Can extend to noncommutative domains (e.g. Ore domains)

This talk : Popov Form

- Hermite Normal Form does not have controlled degrees
  - e.g. degrees of HNF can be larger than input degree
- Popov's form (1969) : purpose was to allow for simple conversion of state space to transfer functions in linear systems theory.
- Villard (1996) introduced Popov form to computer algebra community
- Popov form related to Gröbner bases
- Can extend to noncommutative domains (e.g. Ore domains)
- Question : How to compute (effectively)?

Outline

Definition : Row Popov Form

$$\mathbf{F} = \begin{bmatrix} f_{11} & f_{1,2} & f_{1,3} & \cdots & f_{1,n-1} & f_{1,n} \\ f_{21} & f_{2,2} & f_{2,3} & \cdots & f_{2,n-1} & f_{2,n} \\ f_{31} & f_{3,2} & f_{3,3} & \cdots & f_{3,n-1} & f_{3,n} \\ \vdots & & & & & \\ f_{n-1,1} & f_{n-1,,2} & f_{n-1,3} & \cdots & f_{n-1,n-1} & f_{n-1,n} \\ f_{n,1} & f_{n,,2} & \cdots & \cdots & f_{n,n-1} & f_{n,n} \end{bmatrix}$$

Definition : Row Popov Form

$$\mathbf{F} = \begin{bmatrix} f_{11} & f_{1,2} & f_{1,3} & \cdots & f_{1,n-1} & f_{1,n} \\ f_{21} & f_{2,2} & f_{2,3} & \cdots & f_{2,n-1} & f_{2,n} \\ f_{31} & f_{3,2} & f_{3,3} & \cdots & f_{3,n-1} & f_{3,n} \\ \vdots & & & & & \\ f_{n-1,1} & f_{n-1,,2} & f_{n-1,3} & \cdots & f_{n-1,n-1} & f_{n-1,n} \\ f_{n,1} & f_{n,,2} & \cdots & \cdots & f_{n,n-1} & f_{n,n} \end{bmatrix}$$

- Diagonal entries monic and of row degree

Definition : Row Popov Form

$$\mathbf{F} = \begin{bmatrix} f_{11} & f_{1,2} & f_{1,3} & \cdots & f_{1,n-1} & f_{1,n} \\ f_{21} & f_{2,2} & f_{2,3} & \cdots & f_{2,n-1} & f_{2,n} \\ f_{31} & f_{3,2} & f_{3,3} & \cdots & f_{3,n-1} & f_{3,n} \\ \vdots & & & & & \\ f_{n-1,1} & f_{n-1,,2} & f_{n-1,3} & \cdots & f_{n-1,n-1} & f_{n-1,n} \\ f_{n,1} & f_{n,,2} & \cdots & \cdots & f_{n,n-1} & f_{n,n} \end{bmatrix}$$

- Diagonal entries monic and of row degree
- deg $f_{j,i}$ < deg $f_{i,i}$ for $j \neq i$

Definition : Row Popov Form

$$\mathbf{F} = \begin{bmatrix} f_{11} & f_{1,2} & f_{1,3} & \cdots & f_{1,n-1} & f_{1,n} \\ f_{21} & f_{2,2} & f_{2,3} & \cdots & f_{2,n-1} & f_{2,n} \\ f_{31} & f_{3,2} & f_{3,3} & \cdots & f_{3,n-1} & f_{3,n} \\ \vdots & & & & & \\ f_{n-1,1} & f_{n-1,,2} & f_{n-1,3} & \cdots & f_{n-1,n-1} & f_{n-1,n} \\ f_{n,1} & f_{n,,2} & \cdots & \cdots & f_{n,n-1} & f_{n,n} \end{bmatrix}$$

- Diagonal entries monic and of row degree
- deg $f_{j,i} <$ deg $f_{i,i}$ for $j \neq i$
- deg $f_{i,j} <$ deg $f_{i,i}$ for $j < i$

Definition : Row Popov Form

$$\mathbf{F} = \begin{bmatrix} f_{11} & f_{1,2} & f_{1,3} & \cdots & f_{1,n-1} & f_{1,n} \\ f_{21} & f_{2,2} & f_{2,3} & \cdots & f_{2,n-1} & f_{2,n} \\ f_{31} & f_{3,2} & f_{3,3} & \cdots & f_{3,n-1} & f_{3,n} \\ \vdots & & & & & \\ f_{n-1,1} & f_{n-1,,2} & f_{n-1,3} & \cdots & f_{n-1,n-1} & f_{n-1,n} \\ f_{n,1} & f_{n,,2} & \cdots & \cdots & f_{n,n-1} & f_{n,n} \end{bmatrix}$$

- Diagonal entries monic and of row degree
- deg $f_{j,i} <$ deg $f_{i,i}$ for $j \neq i$
- deg $f_{i,j} <$ deg $f_{i,i}$ for $j < i$
- deg $f_{i,j} \leq$ deg $f_{i,i}$ for $j > i$
- zero rows at bottom

Lots of variations (via reordering).

Definition : Row Popov Form

$$\mathbf{F} = \begin{bmatrix} f_{11} & f_{1,2} & f_{1,3} & \cdots & f_{1,n-1} & f_{1,n} \\ f_{21} & f_{2,2} & f_{2,3} & \cdots & f_{2,n-1} & f_{2,n} \\ f_{31} & f_{3,2} & f_{3,3} & \cdots & f_{3,n-1} & f_{3,n} \\ \vdots & & & & & \\ f_{n-1,1} & f_{n-1,,2} & f_{n-1,3} & \cdots & f_{n-1,n-1} & f_{n-1,n} \\ f_{n,1} & f_{n,,2} & \cdots & \cdots & f_{n,n-1} & f_{n,n} \end{bmatrix}$$

- Diagonal entries monic and of row degree
- $\deg f_{j,i} < \deg f_{i,i}$ for $j \neq i$
- $\deg f_{i,j} < \deg f_{i,i}$ for $j < i$
- $\deg f_{i,j} \leq \deg f_{i,i}$ for $j > i$
- zero rows at bottom

Lots of variations (via reordering).

Example

E.g. : Input degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 3 & 4 & 3 & 3 \\ 4 & 4 & 4 & 4 \\ 6 & 7 & 6 & 7 \end{bmatrix}$$

Output degree bounds for Popov form

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ 2 & 3 & 3 & 7 \end{bmatrix}$$

Alternatively

An polynomial matrix $\mathbf{A}(z)$ is in Popov Form if:

1. it has rank $\mathbf{A}(z)$ non-zero rows;
2. the leading row coefficient is triangular, with monic leading entries;
3. the leading entry of each row has the highest degree in its columns.

Alternatively

An polynomial matrix $\mathbf{A}(z)$ is in Popov Form if:

1. it has rank $\mathbf{A}(z)$ non-zero rows;
2. the leading row coefficient is triangular, with monic leading entries;
3. the leading entry of each row has the highest degree in its columns.

Also called a Polynomial Echelon Form (Kailath book [1980]).

Alternatively

An polynomial matrix $\mathbf{A}(z)$ is in Popov Form if:

1. it has rank $\mathbf{A}(z)$ non-zero rows;
2. the leading row coefficient is triangular, with monic leading entries;
3. the leading entry of each row has the highest degree in its columns.

Also called a Polynomial Echelon Form (Kailath book [1980]).

Any input matrix $\mathbf{A}(z)$ can be transformed into a unique Popov form by row operations.

Popov form as Gröbner Bases

Monomials on vectors $\mathbb{K}^{1\times n}[z]$:

$$z^{\alpha} e_j = [0, \ldots, 0, z^{\alpha}, 0, \ldots, 0]$$

Ordering on monomials of $\mathbb{K}^{1\times n}[z]$:

- Position over Term (POT):

$$z^{\alpha} e_i < z^{\beta} e_j \quad \Longleftrightarrow \quad i < j \quad \text{or } i = j \text{ and } \alpha < \beta$$

Popov form as Gröbner Bases

Monomials on vectors $\mathbb{K}^{1\times n}[z]$:

$$z^{\alpha} e_j = [0, \ldots, 0, z^{\alpha}, 0, \ldots, 0]$$

Ordering on monomials of $\mathbb{K}^{1\times n}[z]$:

- Position over Term (POT):

$$z^{\alpha} e_i < z^{\beta} e_j \quad \Longleftrightarrow \quad i < j \quad \text{or } i = j \text{ and } \alpha < \beta$$

- Term over Position (TOP):

$$z^{\alpha} e_i < z^{\beta} e_j \quad \Longleftrightarrow \quad \alpha < \beta \quad \text{or } \alpha = \beta \text{ and } i < j.$$

Popov form as Gröbner Bases

Monomials on vectors $\mathbb{K}^{1\times n}[z]$:

$$z^\alpha e_j = [0, \ldots, 0, z^\alpha, 0, \ldots, 0]$$

Ordering on monomials of $\mathbb{K}^{1\times n}[z]$:

- Position over Term (POT):

$$z^\alpha e_i < z^\beta e_j \iff i < j \text{ or } i = j \text{ and } \alpha < \beta$$

- Term over Position (TOP):

$$z^\alpha e_i < z^\beta e_j \iff \alpha < \beta \text{ or } \alpha = \beta \text{ and } i < j.$$

If M is a submodule of $\mathbb{K}^{1\times n}[z]$ then we can now speak of Gröbner bases for the module M.

Popov form as Gröbner Bases

(Kojima, Rapisarda, Takaba [System & Control Letters 2007])

Let M be a submodule of $\mathbb{K}^{1\times m}[z]$ with a *term over position* ordering. Then

$\{f_i\}_{i=1,..,s}$ is a reduced Gröbner basis for the module $M \iff$:

(a) $M = \langle f_1, \ldots, f_s \rangle$;

(b) The matrix row$(f_1, \ldots, f_s)$ is in Popov form.

If TOP is replaced by *position over term* ordering then Popov form in (b) is replaced by Hermite form.

Outline

Previous Works

- Popov form algorithm for polynomial matrices:
  - Villard
  - Mulders and Storjohann
  - Beckermann, Labahn, Villard
  - ...

Previous Works

- Popov form algorithm for polynomial matrices:
  - Villard
  - Mulders and Storjohann
  - Beckermann, Labahn, Villard
  - …
- A number of other algorithms for row/column-reduced form of polynomial matrices:
  - Beelen, van den Hurk, Praagman
  - Neven and Praagman
  - …

Previous Works (cont.)

- Elimination-based approaches for Ore Popov form (Giesbrecht, Labahn, Zhang).

Previous Works (cont.)

- Elimination-based approaches for Ore Popov form (Giesbrecht, Labahn, Zhang).
- EG elimination and variants (Abramov, Abramov and Bronstein).

Previous Works (cont.)

- Elimination-based approaches for Ore Popov form (Giesbrecht, Labahn, Zhang).
- EG elimination and variants (Abramov, Abramov and Bronstein).
- The FFreduce algorithm (Beckermann, Cheng, Labahn) computes:

Previous Works (cont.)

- Elimination-based approaches for Ore Popov form (Giesbrecht, Labahn, Zhang).
- EG elimination and variants (Abramov, Abramov and Bronstein).
- The FFreduce algorithm (Beckermann, Cheng, Labahn) computes:
  - a minimal polynomial basis for the left nullspace (in Popov form);

Previous Works (cont.)

- Elimination-based approaches for Ore Popov form (Giesbrecht, Labahn, Zhang).
- EG elimination and variants (Abramov, Abramov and Bronstein).
- The FFreduce algorithm (Beckermann, Cheng, Labahn) computes:
  - a minimal polynomial basis for the left nullspace (in Popov form);
  - GCRD and LCLM (special cases only)

Previous Works (cont.)

- Elimination-based approaches for Ore Popov form (Giesbrecht, Labahn, Zhang).
- EG elimination and variants (Abramov, Abramov and Bronstein).
- The FFreduce algorithm (Beckermann, Cheng, Labahn) computes:
  - a minimal polynomial basis for the left nullspace (in Popov form);
  - GCRD and LCLM (special cases only)
- The FFreduce algorithm is fraction-free.
  i.e. No fractions are introduced while controlling coefficient growth.

Previous Works (cont.)

- Elimination-based approaches for Ore Popov form (Giesbrecht, Labahn, Zhang).
- EG elimination and variants (Abramov, Abramov and Bronstein).
- The FFreduce algorithm (Beckermann, Cheng, Labahn) computes:
    - a minimal polynomial basis for the left nullspace (in Popov form);
    - GCRD and LCLM (special cases only)
- The FFreduce algorithm is fraction-free.
  i.e. No fractions are introduced while controlling coefficient growth.
- A modular algorithm (Cheng, Labahn) for the same computations.

Outline

Method of G. Villard (1996)

- $\mathbf{A}(z)^{-1} = \Delta(z)^{-1}\mathbf{A}^*(z)$ where:
  - $\mathbf{A}^*(z)$ is adjoint of $\mathbf{A}(z)$
  - $\Delta(z)$ is diagonal matrix with det $\mathbf{A}(z)$ on diagonals.
- $\mathbf{A}^*(z)\mathbf{A}(z) = \Delta(z)$ and $\mathbf{A}^*(z) \cdot I = \mathbf{A}^*(z)$ so :
  - $\mathbf{A}^*(z)$ is a gcld of $\Delta(z)$ and $\mathbf{A}^*(z)$.
  - All other gcld's $\mathbf{G}(z)$ are then multiples, i.e.

$$\mathbf{G}(z) = \mathbf{A}^*(z)\mathbf{V}(z) \text{ with } \mathbf{V}(z) \text{ unimodular}$$

Method of G. Villard (1996)

- $\mathbf{A}(z)^{-1} = \Delta(z)^{-1}\mathbf{A}^*(z)$
- If $\mathbf{A}(z)^{-1} = D(z)^{-1}N(z)$ with $D(z)$ of minimal determinant degree in Popov form then

$$D(z) = \mathbf{G}(z)^{-1}\Delta(z) = \mathbf{V}(z)^{-1}\mathbf{A}^*(z)^{-1}\Delta(z) = \mathbf{U}(z)\mathbf{A}(z)$$

  with $\mathbf{U}(z)$ unimodular.
- Therefore find a minimal realization of $\mathbf{A}(z)^{-1}$ having a denominator in Popov form.
- Algorithm exists for the above computation.
- Good for parallel computation

Outline

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 3 & 4 & 3 & 3 \\ 4 & 4 & 4 & 4 \\ 6 & 7 & 6 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & \color{green}{3} & 3 & 3 \\ 3 & 3 & 3 & \color{green}{4} \\ \color{green}{4} & 4 & 4 & 4 \\ 6 & 6 & \color{green}{7} & 7 \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 4 & 4 & 4 & 4 \\ 6 & 7 & 6 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & 3 & 3 & 3 \\ 3 & 2 & 3 & 4 \\ 4 & 4 & 4 & 4 \\ 6 & 6 & 7 & 7 \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ \color{red}{2} & 4 & 4 & 4 \\ 6 & 7 & 6 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & \color{green}{3} & 3 & 3 \\ 3 & 2 & 3 & \color{green}{4} \\ \color{green}{4} & \color{red}{2} & 4 & 4 \\ 6 & 6 & \color{green}{7} & 7 \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ 6 & 7 & 6 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & 3 & 3 & 3 \\ 3 & 2 & 3 & 4 \\ 4 & 2 & 4 & 3 \\ 6 & 6 & 7 & 7 \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ {\color{red}2} & 7 & 6 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & {\color{green}3} & 3 & 3 \\ 3 & 2 & 3 & {\color{green}4} \\ {\color{green}4} & 2 & 4 & 3 \\ 6 & {\color{red}2} & {\color{green}7} & 7 \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ {\color{blue}5} & {\color{red}3} & 6 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & {\color{green}3} & 3 & 3 \\ 3 & 2 & 3 & {\color{green}4} \\ {\color{green}4} & 2 & 4 & 3 \\ 6 & {\color{blue}5} & {\color{green}7} & {\color{red}3} \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ \color{red}{2} & \color{blue}{5} & 6 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & \color{green}{3} & 3 & 3 \\ 3 & 2 & 3 & \color{green}{4} \\ \color{green}{4} & 2 & 4 & 3 \\ 6 & \color{red}{2} & \color{green}{7} & \color{blue}{5} \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ {\color{blue}4} & {\color{blue}5} & {\color{red}3} & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & {\color{green}3} & 3 & 3 \\ 3 & 2 & 3 & {\color{green}4} \\ {\color{green}4} & 2 & 4 & 3 \\ {\color{red}3} & {\color{blue}4} & {\color{green}7} & {\color{blue}5} \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ {\color{red}2} & {\color{blue}5} & 3 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & {\color{green}3} & 3 & 3 \\ 3 & 2 & 3 & {\color{green}4} \\ {\color{green}4} & 2 & 4 & 3 \\ 3 & {\color{red}2} & {\color{green}7} & {\color{blue}5} \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ {\color{blue}3} & {\color{red}3} & {\color{blue}4} & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & {\color{green}3} & 3 & 3 \\ 3 & 2 & 3 & {\color{green}4} \\ {\color{green}4} & 2 & 4 & 3 \\ {\color{red}3} & {\color{blue}4} & {\color{green}7} & {\color{red}3} \end{bmatrix}$$

Mulders-Storjohann Procedure

First transform $\mathbf{A}(z)$ to *Weak Popov Form* - basically where pivots are on seperate rows but nothing more. Then convert to Popov Form

E.g. : degree bounds (and so on ..)

$$\begin{bmatrix} 3 & 3 & 2 & 3 \\ 2 & 4 & 3 & 3 \\ 2 & 3 & 4 & 4 \\ 2 & 3 & 3 & 7 \end{bmatrix} \text{ or } \begin{bmatrix} 2 & 3 & 3 & 3 \\ 3 & 2 & 3 & 4 \\ 4 & 2 & 4 & 4 \\ 3 & 2 & 7 & 3 \end{bmatrix}$$

Outline

Symbolic Domains

- Basic coefficient domain: Quotient field: $\mathbb{F}(\alpha_1, \ldots, \alpha_k)$
  - symbols are first class objects in CA environments.
- Polynomial arithmetic easier than arithmetic with rational functions

$$\frac{a(x)}{b(x)} + \frac{c(x)}{d(x)} = \frac{a(x) \cdot d(x) + b(x) \cdot c(x)}{b(x) \cdot d(x)}$$

  Need to rcognize 0 : need to normalize out gcd's at every step
- Basic goal:

  To work with polynomial arithmetic in integral domain (e.g. in $\mathbb{F}[\alpha_1, \ldots, \alpha_k]$) rather than in quotient field.
- Want to do our arithmetic **fraction-free** but at the same time to minimize growth of intermediate computation.

Symbolic Domains

$$A = \begin{bmatrix} a & b & c & \cdots & \cdots \\ d & e & f & \cdots & \cdots \\ g & h & i & \cdots & \cdots \\ \vdots & \vdots & \vdots & & \end{bmatrix} \approx \begin{bmatrix} a & b & c & \cdots & \cdots \\ 0 & \tilde{e} & \tilde{f} & \cdots & \cdots \\ 0 & \tilde{h} & \tilde{i} & \cdots & \cdots \\ \vdots & \vdots & \vdots & & \end{bmatrix}$$

- Cross multiplication gives exponential growth of coeffs
- Fraction-free Gaussian elimination (FFGE)

$$A \approx \begin{bmatrix} a & b & c & \cdots & \cdots \\ 0 & \tilde{e} & \tilde{f} & \cdots & \cdots \\ 0 & 0 & a(..) & \cdots & a(...) \\ \vdots & \vdots & \vdots & & \\ 0 & 0 & a(..) & \cdots & a(...) \end{bmatrix}.$$

Allows for linear growth of coefficient size.

- Important : computes Cramer solution of linear problem.

Popov Form via Order Basis

- $\mathbf{U}(z)\mathbf{A}(z) = \mathbf{T}(z)$ same as $[\mathbf{U}(z), \mathbf{T}(z)] \begin{bmatrix} \mathbf{A}(z) \\ -I_n \end{bmatrix} = 0$
- $\mathbf{U}(z)\mathbf{A}(z) = \mathbf{T}(z)$ same as $[\mathbf{U}(z), \mathbf{T}(z)z^{\text{vec}r}] \begin{bmatrix} \mathbf{A}(z)z^{\text{vec}r} \\ -I_n \end{bmatrix} = 0$

  for any vector vec$r$.
- Choose vec$r$ intelligently so that $[\mathbf{U}(z), \mathbf{T}(z)z^{\text{vec}r}]$ has leading coefficient the same as leading coefficient of $[0, \mathbf{T}(z)]$.
- Find Popov form for $[\mathbf{U}(z), \mathbf{T}(z)z^{\text{vec}r}]$

Popov Form via Order Basis

- $\mathbf{U}(z)\mathbf{A}(z) = \mathbf{T}(z)$ same as $[\mathbf{U}(z), \mathbf{T}(z)] \left[\begin{array}{c} \mathbf{A}(z) \\ -I_n \end{array} \right] = 0$
- $\mathbf{U}(z)\mathbf{A}(z) = \mathbf{T}(z)$ same as $[\mathbf{U}(z), \mathbf{T}(z)z^{\text{vec}r}] \left[\begin{array}{c} \mathbf{A}(z)z^{\text{vec}r} \\ -I_n \end{array} \right] = 0$ for any vector vec$r$.
- Choose vec$r$ intelligently so that $[\mathbf{U}(z), \mathbf{T}(z)z^{\text{vec}r}]$ has leading coefficient the same as leading coefficient of $[0, \mathbf{T}(z)]$.
- Find Popov form for $[\mathbf{U}(z), \mathbf{T}(z)z^{\text{vec}r}]$

Works because we can use *order bases* to solve last problem.

Good because order basis computation can be done via fraction-free methods (FFGE method of Beckermann-Labahn)

Popov Form via Order Basis (cont.)

- Order basis finds a module basis for problem:

$$f_1(z)m_1(z) + \cdots + f_n(z)m_n(z) = O(z^\sigma)$$

Popov Form via Order Basis (cont.)

- Order basis finds a module basis for problem:

$$f_1(z)m_1(z) + \cdots + f_n(z)m_n(z) = O(z^{\sigma})$$

- Order basis is form of an $n \times n$ matrix polynomial

Popov Form via Order Basis (cont.)

- Order basis finds a module basis for problem:

$$f_1(z)m_1(z) + \cdots + f_n(z)m_n(z) = O(z^{\sigma})$$

- Order basis is form of an $n \times n$ matrix polynomial
- FFGE computes order basis in a *shifted* Popov Form using fraction-free arithmetic

Popov Form via Order Basis (cont.)

- Order basis finds a module basis for problem:

$$f_1(z)m_1(z) + \cdots + f_n(z)m_n(z) = O(z^{\sigma})$$

- Order basis is form of an $n \times n$ matrix polynomial
- FFGE computes order basis in a *shifted* Popov Form using fraction-free arithmetic
- Choose vector vec$r$ intelligently (use adjoint of $\mathbf{A}(z)$) so that one can embed Popov computational inside

$$\begin{bmatrix} \mathbf{M}_{11}(z) & \mathbf{M}_{12}(z) \\ \mathbf{M}_{21}(z) & \mathbf{M}_{22}(z) \end{bmatrix} \begin{bmatrix} \mathbf{A}(z) z^{\text{vec}r} \\ -I_n \end{bmatrix} = \begin{bmatrix} \mathbf{R}(z) z^{\text{vec}\sigma} \\ 0 \end{bmatrix}$$

Popov Form via Order Basis (cont.)

- Order basis finds a module basis for problem:

$$f_1(z)m_1(z) + \cdots + f_n(z)m_n(z) = O(z^{\sigma})$$

- Order basis is form of an $n \times n$ matrix polynomial
- FFGE computes order basis in a *shifted* Popov Form using fraction-free arithmetic
- Choose vector vec$r$ intelligently (use adjoint of $\mathbf{A}(z)$) so that one can embed Popov computational inside

$$\begin{bmatrix} \mathbf{M}_{11}(z) & \mathbf{M}_{12}(z) \\ \mathbf{M}_{21}(z) & \mathbf{M}_{22}(z) \end{bmatrix} \begin{bmatrix} \mathbf{A}(z) z^{\text{vec}\, r} \\ -I_n \end{bmatrix} = \begin{bmatrix} \mathbf{R}(z) z^{\text{vec}\, \sigma} \\ 0 \end{bmatrix}$$

Future Topics

1. Want fraction-free reduction procedure

Future Topics

1. Want fraction-free reduction procedure
2. Relationship of Popov Form (and its computation) to work of Pryce [2001] with Taylor series for numerical solution of DAEs

Future Topics

1. Want fraction-free reduction procedure
2. Relationship of Popov Form (and its computation) to work of Pryce [2001] with Taylor series for numerical solution of DAEs
3. Higher order methods for systems of linear odes without conversion to first order systems

Future Topics

1. Want fraction-free reduction procedure
2. Relationship of Popov Form (and its computation) to work of Pryce [2001] with Taylor series for numerical solution of DAEs
3. Higher order methods for systems of linear odes without conversion to first order systems
4. Involve adjoint calculation in process.

Future Topics

1. Want fraction-free reduction procedure
2. Relationship of Popov Form (and its computation) to work of Pryce [2001] with Taylor series for numerical solution of DAEs
3. Higher order methods for systems of linear odes without conversion to first order systems
4. Involve adjoint calculation in process.
   - Did this in case of Order Basis (B & L, submitted to ISSAC 2009)