>

Comparisons of of timings between the Gröbner bases (GB) method
and the linear algebra (LA) method from our paper:

On Rational Recursion for Holonomic Sequences

Bertrand Teguia Tabuguia and James Worrell
Department of Computer Science, University of Oxford

The summary is given on Table 1 in the paper.

```
> restart
> with(NLDE, HoloToSimpleRatrec, OrderDegreeRec) ;
      #we import our procedure HoloToSimpleRatrec from the package NLDE;
  #OrderDegreeRec returns the order and the degree of  an algebraic recurrence equation
```

$$[HoloToSimpleRatrec, OrderDegreeRec] \quad (1)$$

```
> with(CodeTools, CPUTime); #we import the command CPUTime to evalute the timings
```

$$[CPUTime] \quad (2)$$

>

Random functions for the orders and the degrees

```
> randorder := rand(6..10) :
> randdegree := rand(1..5) :
>
```

Generating holonomic equations of orders $\leq$ 10 and 1 $\leq$ degrees $\leq$ 5

```
> r1 := randorder( )
```

$$r1 := 10 \quad (1.1)$$

```
> d1 := randdegree( )
```

$$d1 := 2 \quad (1.2)$$

```
> RE1 := add(randpoly(n, degree = d1, coeffs = rand(-1..1)) · s(n + j), j = 0..r1) = 0 :
> r2 := randorder( )
```

$$r2 := 9 \quad (1.3)$$

```
> d2 := randdegree( )
```

$$d2 := 3 \quad \textbf{(1.4)}$$

> $RE2 := add(randpoly(n, degree = d2, coeffs = rand(-1..1)) \cdot s(n+j), j = 0..r2) = 0 :$

> $r3 := randorder(\,)$

$$r3 := 8 \quad \textbf{(1.5)}$$

> $d3 := randdegree(\,)$

$$d3 := 4 \quad \textbf{(1.6)}$$

> $RE3 := add(randpoly(n, degree = d3, coeffs = rand(-1..1)) \cdot s(n+j), j = 0..r3) = 0 :$

> $r4 := randorder(\,)$

$$r4 := 6 \quad \textbf{(1.7)}$$

> $d4 := randdegree(\,)$

$$d4 := 5 \quad \textbf{(1.8)}$$

> $RE4 := add(randpoly(n, degree = d4, coeffs = rand(-1..1)) \cdot s(n+j), j = 0..r4) = 0 :$

> $r5 := randorder(\,)$

$$r5 := 7 \quad \textbf{(1.9)}$$

> $d5 := randdegree(\,)$

$$d5 := 1 \quad \textbf{(1.10)}$$

> $RE5 := add(randpoly(n, degree = d5, coeffs = rand(-1..1)) \cdot s(n+j), j = 0..r5) = 0 :$

>

>

Selected holonomic equations

> $RE1$

$$-n^2 s(n) + (n^2-1) s(n+1) + (n^2-n-1) s(n+2) + n\, s(n+3) + (n+1) s(n+4) + (-n^2-1) s(n+5) - n^2 s(n+6) + (n^2+n) s(n+7) + (-1-n) s(n+8) + (n^2-n) s(n+9) + (-n^2+1) s(n+10) = 0 \quad \textbf{(2.1)}$$

> $RE2$

$$(-n^2+n+1) s(n) + (n^3-n^2+n+1) s(n+1) + (n^2-n-1) s(n+2) + (n^3-n^2-n) s(n+3) + n\, s(n+4) + (n^3-n-1) s(n+5) + (-n^3+n^2-n+1) s(n+6) + (n^2-n) s(n+7) + (-n^3+n^2-n+1) s(n+8) + (-n^3-n^2) s(n+9) = 0 \quad \textbf{(2.2)}$$

> $RE3$

$$(n^4-n^3-n^2-1) s(n) + (n^4-n^2-n-1) s(n+1) + (-n^4+n^3-n^2+n) s(n+2) + (n^2+1) s(n+3) + (-n^4-n^3+n^2+n+1) s(n+4) + (-n^4+n^2+n+1) s(n+5) + (-n^4+n^3+n^2) s(n+6) + (-n^4+n^2+n) s(n+7) + (n^3+n^2-n) s(n+8) = 0 \quad \textbf{(2.3)}$$

> $RE4$

$$(-n^4+n^2-n+1) s(n) + (n^4-n^3+n^2+n) s(n+1) + (-n^5-n^4+n^2+n) s(n+2) + (n^4+n^2-1) s(n+3) + (-n^4-n^2-n+1) s(n+4) + (-n^4-n^3+n^2 \quad \textbf{(2.4)}$$

$-n-1)\, s(n+5)+\left(-n^3+n^2+n-1\right) s(n+6)=0$

```
> RE5
```

$$(-1-n)\, s(n)-n\, s(n+1)+(-n+1)\, s(n+2)+(-1-n)\, s(n+3)+(n+1)\, s(n+4)-n\, s(n+5)+s(n+6)+(-1-n)\, s(n+7)=0 \quad (2.5)$$

```
>
```

GB method

```
> t_gbeq1, GBeq1 := CPUTime(timelimit(300, HoloToSimpleRatrec(RE1, s(n), method = GB))) : t_gbeq1
```

$$0.218 \quad (3.1)$$

```
> t_gbeq2, GBeq2 := CPUTime(timelimit(300, HoloToSimpleRatrec(RE2, s(n), method = GB))) : t_gbeq2
```

$$27.547 \quad (3.2)$$

```
> t_gbeq3, GBeq3 := CPUTime(timelimit(300, HoloToSimpleRatrec(RE3, s(n), method = GB))) : t_gbeq3
Error, (in libmgb:-gbasis) time expired
```

$$t\_gbeq3 \quad (3.3)$$

```
> t_gbeq4, GBeq4 := CPUTime(timelimit(300, HoloToSimpleRatrec(RE4, s(n), method = GB))) : t_gbeq4
Error, (in libmgb:-gbasis) time expired
```

$$t\_gbeq4 \quad (3.4)$$

```
> t_gbeq5, GBeq5 := CPUTime(timelimit(300, HoloToSimpleRatrec(RE5, s(n), method = GB))) : t_gbeq5
```

$$0. \quad (3.5)$$

```
>
>
```

LA method

```
> t_laeq1, LAeq1 := CPUTime(HoloToSimpleRatrec(RE1, s(n), method = LA)) : t_laeq1
```

$$0.031 \quad (4.1)$$

```
> t_laeq2, LAeq2 := CPUTime(HoloToSimpleRatrec(RE2, s(n), method = LA)) : t_laeq2
```

$$0.188 \quad (4.2)$$

```
> t_laeq3, LAeq3 := CPUTime(HoloToSimpleRatrec(RE3, s(n), method = LA)) : t_laeq3
```

$$0.734 \quad (4.3)$$

```
> t_laeq4, LAeq4 := CPUTime(HoloToSimpleRatrec(RE4, s(n), method = LA)) : t_laeq4
```

$$2.203 \quad (4.4)$$

```
> t_laeq5, LAeq5 := CPUTime(HoloToSimpleRatrec(RE5, s(n), method = LA)) : t_laeq5
```

$$0. \quad (4.5)$$

```
>
>
```

Observation: in avarage the LA method is often faster than the GB method as the two list of timings below illustrates. We also provide a list with the orders and degrees of the output simple ratrec equations.

> $GBtimings := [t\_gbeq1, t\_gbeq2, t\_gbeq3, t\_gbeq4, t\_gbeq5]$

$$GBtimings := [0.218, 27.547, t\_gbeq3, t\_gbeq4, 0.] \tag{3}$$

> $LAtimings := [t\_laeq1, t\_laeq2, t\_laeq3, t\_laeq4, t\_laeq5]$

$$LAtimings := [0.031, 0.188, 0.734, 2.203, 0.] \tag{4}$$

> $GBoutputs := [OrderDegreeRec(GBeq1, s(n)), OrderDegreeRec(GBeq2, s(n)), None, None, OrderDegreeRec(GBeq5, s(n))]$

$$GBoutputs := [[12, 3], [11, 5], None, None, [8, 2]] \tag{5}$$

> $LAoutputs := [OrderDegreeRec(LAeq1, s(n)), OrderDegreeRec(LAeq2, s(n)), OrderDegreeRec(LAeq3, s(n)), OrderDegreeRec(LAeq4, s(n)), OrderDegreeRec(LAeq5, s(n))]$

$$LAoutputs := [[12, 3], [12, 4], [12, 5], [11, 6], [8, 2]] \tag{6}$$

>