UTPA The University of Texas-Pan American

Michael Brazier, Arthur D. Chtcherba
Department of Computer Science
The University of Texas - Pan American
Edinburg, Texas, USA

Computing Determinants of Symbolic Matrices using a Straight Line Program Interpolator

Deepak Kapur
Department of Computer Science
The University of New Mexico
Albuquerque, New Mexico, USA

University of New Mexico

Abstract

We present a software package for computing determinants of matrices with polynomial entries using an interpolator based on straight line programs (SLPs). One of the applications of this tool is computation of multivariate resultants. The tool enabled computing the determinant of a Dixon dialytic matrix, a multivariate resultant matrix based on the Dixon-Cayley formulation, for a PDE stability problem[5] presented by Hong at ACA 2004, from which the resultant can be extracted by factorization.

The software allows expressing interpolating polynomials as SLPs. Besides standard arithmetic operations (+,-,*,/,^), the determinant of a matrix with polynomial entries is considered an SLP statement. In its first release, the tool used Zippel's sparse probabilistic algorithm which interpolated one variable at a time with a few improvements. Zipple's algorithm has now been generalized to allow interpolation of many variables in a single stage. Recently, the tool has been expanded to allow interpolation of more general SLP statements, including implementation of interpolation algorithms by Kaltofen at al in which terms are pruned using a homogenizing variable.

An Maple interface for translating matrices with polynomial entries into a format acceptable by the package is available as well.

Applications

- Symbolic Matrix determinant extraction
  - Computations of Projections operators (resultants)
- General Black Box (specified as SLP) interpolator
- Avoids Intermediate expression swell in algebraic computations

Software Features

- Sparse probabilistic interpolation [3]
- Ability to specify arbitrary size field
- Automatic degree and sparseness estimation
- Fast Fourier Transform [2]
- Permanent term pruning [1]
- Interpolation strategies
- Compatibility with Maple
- Multiplatform: Linux/Windows/Mac

Heuristics

- Interpolation strategy selection
  - Variable Order
- Fast Fourier Transform vs. Classic
- Univariate Vandermonde vs. Newton

Future Work

- Software Improvements
  - Improvements to Finite Field arithmetic
  - X86 assembly implementation
  - Cashing values for subexpresions
- Black Box compiler
- Parallel computing
  - Multiprocessor Support
  - Distributed interpolation
- Algorithms
  - Factorized interpolation [4]
  - Temporary Pruning [1]
  - Galois Field arithmetic
  - Interpolation Strategies Heuristics

References

[1] Erich Kaltofen and Wen-shin Lee. Early termination in sparse interpolation algorithms. *J. Symbolic Comput.*, 36(3-4):365-400, 2003. Special issue (ISSAC 2002)..

[2] E. Kaltofen and Lakshman Yagati. Improved sparse multivariate polynomial interpolation algorithms. In *ISSAC '88 Proc.*, pages 467-474.

[3] Zippel, R. Interpolating polynomials from their values. *J. Symbolic Comput.* 9, 3 (1990), 375-403.

[4] E. Kaltofen and B. Trager. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *J. Symbolic Comput.*, 9(3):301-320, 1990.

[5] Hong, H., Liska, R., and Steinberg, S. Testing stability by quantifier elimination. *J. Symb. Comput.* 24, 2 (Aug. 1997), 161-187.

Input File

```
RunProfile:=table([RNGSeed=15,Primes=[3221225473]]);

Problem:=proc(A,a,b,c)
  # Dixon Matrix constructed for Herons Formula,
  Elimination:=array(`sparse`,1..7,1..7,[
      (1,1)=a^2,(1,2)=2*A, (1,3)=2*A,  (1,4)=-a*c^2, (1,5)=-a*b^2,
      (2,1)=-a, (2,4)=a^2, (2,6)=2*A,  (2,7)=-a*b^2,
      (3,1)=-a, (3,5)=a^2, (3,6)=2*A,  (3,7)=-a*c^2,
      (4,2)=-a, (4,3)=-a,  (4,4)=2*A,  (4,5)=2*A,
      (5,6)=-a, (5,7)=2*A, (6,6)=-a,   (6,7)=2*A,
      (7,4)=-a, (7,5)=-a,  (7,7)=a^2
  ]);
  return det(Elimination);
end proc;

Polynomial:=interpolate(Problem, timeLimit=3600.000);
```

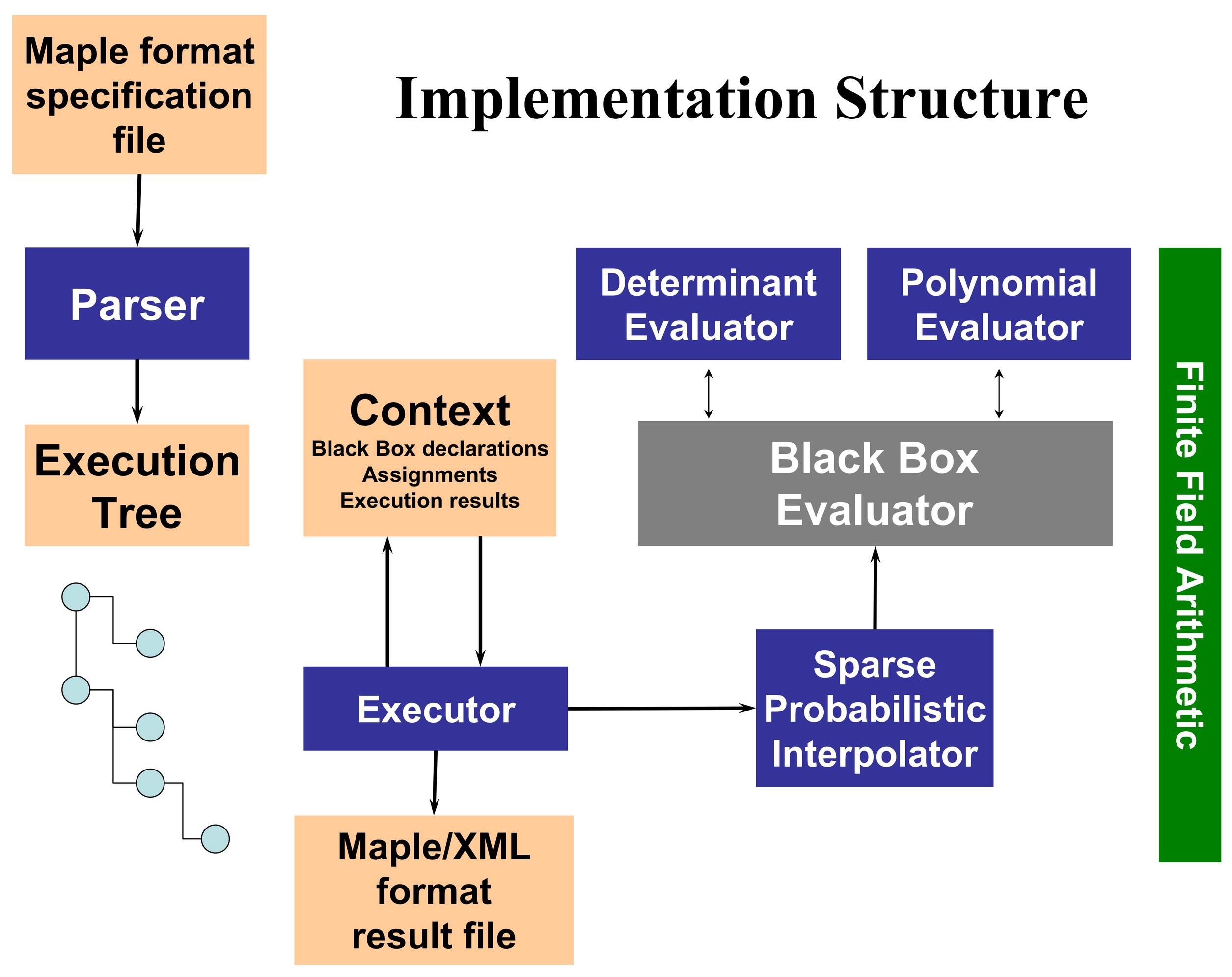
Screen Output

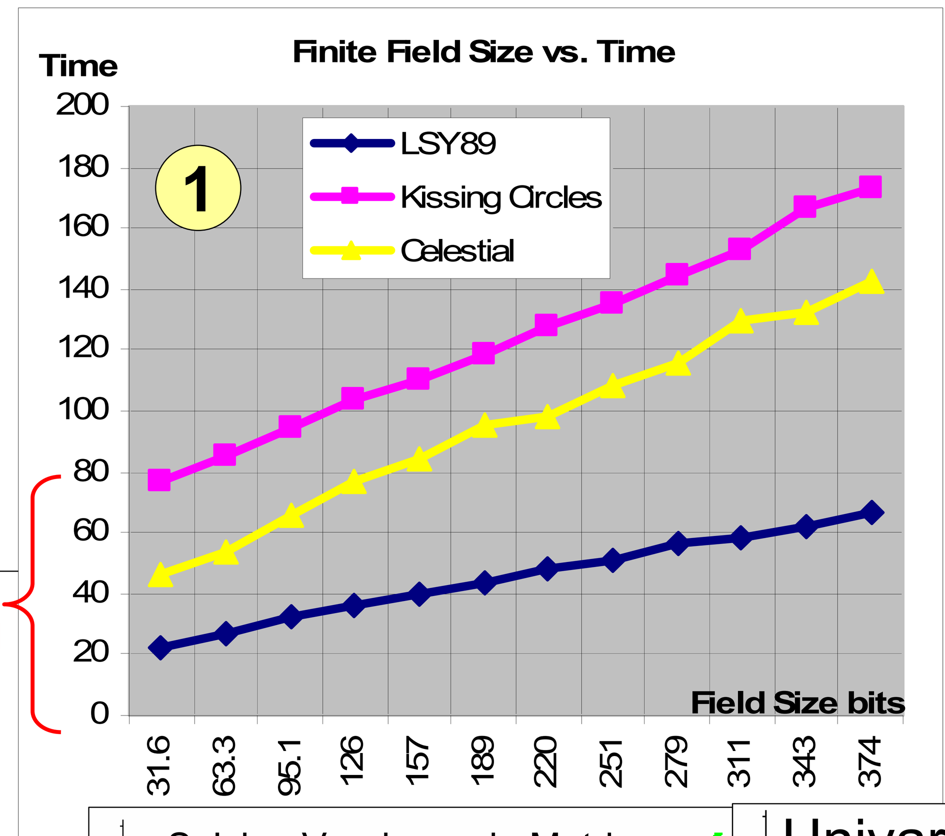
```
Parsing file "sample-input.txt"...Output sent to file "sample.txt".

Field operations: 2.11406e+6 adds/s, 4.36907e+6 mults/s
Set up n*n Vandermonde matrix:
        quad = 3.05e-8*n^2+1.5e-6*n
        FFT = 8.54492e-6*n*log[2](n)^2
Solve n*n Vandermonde matrix:
        quad = 4.69e-7*n^2-1.6e-5*n +0
        FFT = 8.23242e-6*n*log[2](n)^2
Switching to FFT Vandermonde solver at 2197 terms.

Univariate, Newton(d) = 4.145e-7*d^2 -8.5e-6*d
Univariate, Vandermonde(t) = time to solve t*t Vandermonde matrix

Interpolation time limit: 3600 sec (0d 1h 0m 0s)
Evaluations/sec of expression to interpolate: 16129
Estimated evaluation/Vandermonde crossover at 153 terms.

Tightening degree bounds:
Variable Est Time Evals     Time Min Deg Max Deg Terms
-------- -------- ----- -------- ------- ------- -----
       A      0.0     6      0.0       0       2     2
       a      0.0    11      0.0       6      10     3
       b      0.0     5      0.0       0       4     3
       c      0.0     5      0.0       0       4     3
-------- -------- ----- -------- ------- ------- -----
                     27      0.0

Maximum number of terms: 54
Variables interpolate in order: a, b, c, A
                                       Stage breakdown:               Polynomial
Variable No Pts (s)Est Evals (s)Stage (s)Eval (s)Vand (s)Intrp Terms Deg
-------- -- --- ------ ----- -------- ------- ------- -------- ----- ---
       a  1  11    0.0     0      0.0     0.0     0.0      0.0     3  10
       b  2   5    0.0    12      0.0     0.0     0.0      0.0     6  10
       c  3   5    0.0    24      0.0     0.0     0.0      0.0     7  10
       A  4   3    0.0    14      0.0     0.0     0.0      0.0     7  10
-------- -- --- ------ ----- -------- ------- ------- -------- ----- ---
                          50      0.0
```

Output File

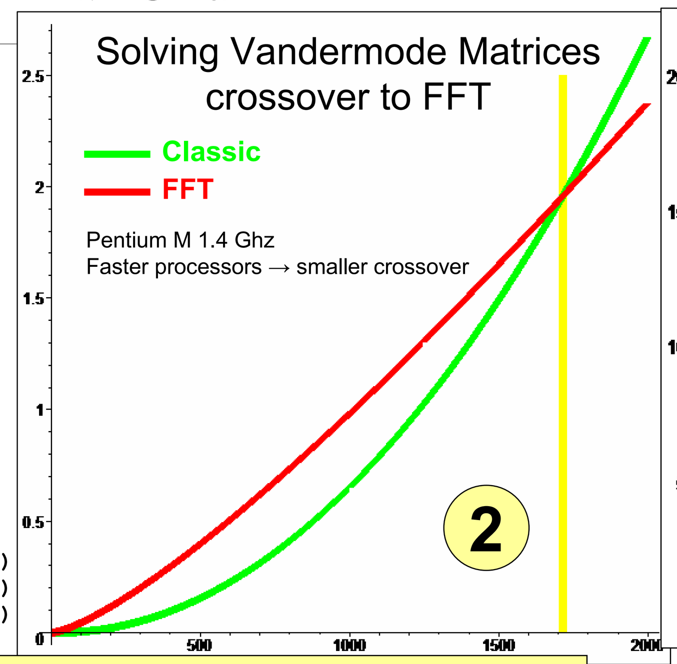
```
# Start execution: 11:36:25 Aug-30-2005
# Input file: "..\sample-input.txt"
# Polynomial data:
Benchmark := table([
  field = table([add = 4.73022e-7, mult = 2.28882e-7]),
  quadVand = table([setup = 3.05e-8*n^2 +1.5e-6*n,
                    solve = 4.69e-7*n^2 -1.6e-5*n]),
  FFTVand = table([
    setup = 8.54492e-006*n*log[2](n)^2,
    solve = 8.23242e-006*n*log[2](n)^2
  ]),
  adaptCross = 2197.47,  Newton = 4.145e-7*d^2 -8.5e-6*d]);
Polynomial := table([
  errorProbability = 6.084641e-006,
  terms = 7, totalDegree = 10,
  degrees = [10, 4, 4, 2],
  value = 16*a^6*A^2+a^6*c^4-2*a^6*b^2*c^2+a^6*b^4-2*a^8*c^2-2*a^8*b^2+a^10
]);
_Stats_ := table([
  Polynomial = table([
    evalRate = 16129, totalEvals = 50, totalTime = 0,
    bounds = table([
      a = table([evals = 11, estTime = 0.000638655, degree = 10, terms = 3]),
      b = table([evals = 5,  estTime = 0.000277863, degree = 4,  terms = 3]),
      c = table([evals = 5,  estTime = 0.000277863, degree = 4,  terms = 3]),
      A = table([evals = 6,  estTime = 0.000335922, degree = 2,  terms = 2])
    ]),
    stages = table([
      a = table([evals = 0,  polyTerms = 3, polyDegree = 10    ]),
      b = table([evals = 12, estTime = 0.000477246, polyTerms = 6, polyDegree = 10])
      c = table([evals = 24, estTime = 0.000988809, polyTerms = 7, polyDegree = 10])
      A = table([evals = 14, estTime = 0.00054957,  polyTerms = 7, polyDegree = 10])
    ])
  ]),
  Problem = table([
   retVal = table([look = 24, add = 10,
     neg = 10, mul = 48, inv = 4,
     arguments = table([
       rank = 6, density = 0.489796,
       minorRows = [1, 2, 3, 4, 5, 7],
       minorCols = [1, 2, 4, 5, 6, 7]
     ])
   ])
  ])
]);
Status := 'SUCCESS';
totalTime := 0.031000;
ate := ["2005-08-30", "11:36:32"];
# End execution: 11:36:32 Aug-30-2005
```

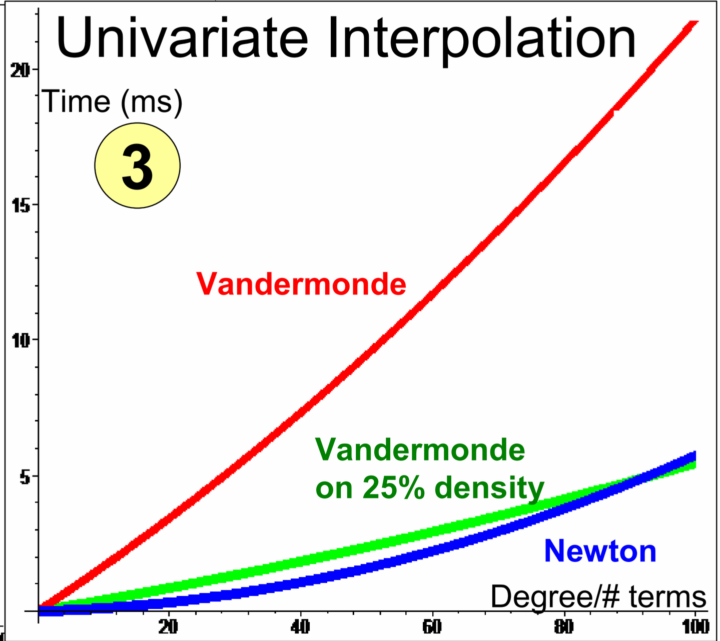
Implementation Structure





Field Overhead





Variable	No	Pts	(s)Est	Evals	(s)Stage	(s)Eval	(s)Vand	(s)Intrp	Polynomial Terms	Polynomial Deg
u_y	1	11	0.0	0	0.0	0.0	0.0	0.0	11	10
u_x	2	11	0.1	110	0.1	0.1	0.0	0.0	65	10
r_y	3	11	0.4	650	0.4	0.3	0.0	0.0	600	16
r_x	4	11	6.2	6000	5.8	3.3	2.4	0.1	3051	16
q_x	5	11	68.5	30510	64.8	18.0	46.5	0.3	18757	22
p_y	6	11	591.6	187570	644.7	123.4	519.9	1.4	46700	31
p_x	7	11	1635.5	467000	2038.4	338.1	1696.4	3.9	183796	31
q_y	8	11	7840.3	1837960	18592.0	2137.1	16436.7	18.2	183796	31
t_x	9	10	7118.3	1654164	16745.8	1897.9	14832.8	15.1	183796	32
Classic				4183964	38091.9					

4

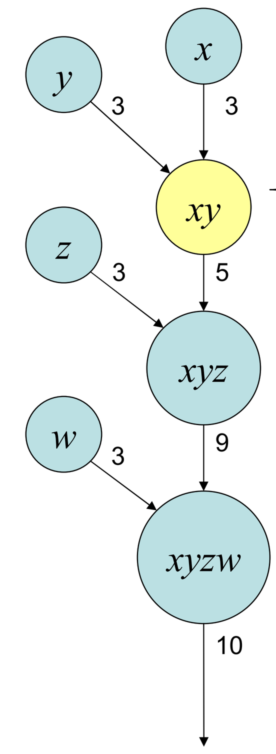
Variable	No	Pts	(s)Est	Evals	(s)Stage	Poly	Eval	Vand	Intrp	Poly Terms Filled	Partial
all	0	1	0.0	0	0.0	0.0	0.0	0.0	0.0	0	1
u_x	1	11	0.0	10	0.0	0.0	0.0	0.0	0.0	0	11
r_y	2	11	0.1	110	0.1	0.0	0.0	0.0	0.0	0	110
r_x	3	11	0.8	1100	0.7	0.0	0.6	0.1	0.0	0	605
q_x	4	11	6.1	6050	5.8	0.0	3.3	2.4	0.1	0	3950
p_y	5	11	87.4	39500	87.5	0.0	25.2	61.9	0.4	0	13226
p_x	6	11	369.4	132260	393.7	0.0	87.1	305.5	1.0	0	43727
q_y	7	11	1513.3	437270	1952.0	0.1	349.6	1597.9	4.5	0	183796
t_x	8	10	7038.7	1654164	16945.2	0.4	1936.3	14994.4	14.1	0	183796
u_y	9	11	0.0	0	0.0	0.0	0.0	0.0	0.0	183796	0
With Pruning				2270464	19384.9						

1. Finite Field arithmetic has large overhead
2. FFT is faster than classic even on small examples
3. Vandermonde Univariate interpolation is faster than Newton even at 25% density
4. Permanent pruning technique [1] effective even when no pruning happens on homogeneous cases
5. In many cases, interpolating multiple variables per stage is faster. Better strategies are necessary to choose interpolation tree.

Sponsor: National Science Foundation
Grant: NSF grant no. CCR-0203051
University: University of New Mexico
Department: Department of Computer Science, Dr. D. Kapur

Classic Zippels Algorithm

Interpolating $f(x,y,z,w)$



$-27311\,x - 3191\,x^3 + 54$ $y = 5, z = 7, w = 11$

$x^3, x, 1$ skeleton

Solving Vandermonde Matrices
- Classic Quadratic Algorithm
- Fast Fourier Transform
- (Adaptive Picks best)

$$\begin{pmatrix} x^3 & x & 1 \\ x^6 & x^2 & 1 \\ x^9 & x^3 & 1 \end{pmatrix}$$

1. Evaluate Black Box 3 times to
2. Solve Vandermonde Matrix to
3. Obtain 1 one value for each coefficient
4. Repeat 3 times (max degree of y + 1)

$$\left(\begin{pmatrix} x^3 & x & 1 \\ x^6 & x^2 & 1 \\ x^9 & x^3 & 1 \end{pmatrix}^{-1} \times \begin{pmatrix} f(3,5,7,11) & f(3,-1,7,11) & f(3,-4,7,11) \\ f(3^2,5,7,11) & f(3^2,-1,7,11) & f(3^2,-4,7,11) \\ f(3^3,5,7,11) & f(3^3,-1,7,11) & f(3^3,-4,7,11) \end{pmatrix}\right)^T$$

Max Degree of y is 2, → 3 eval points

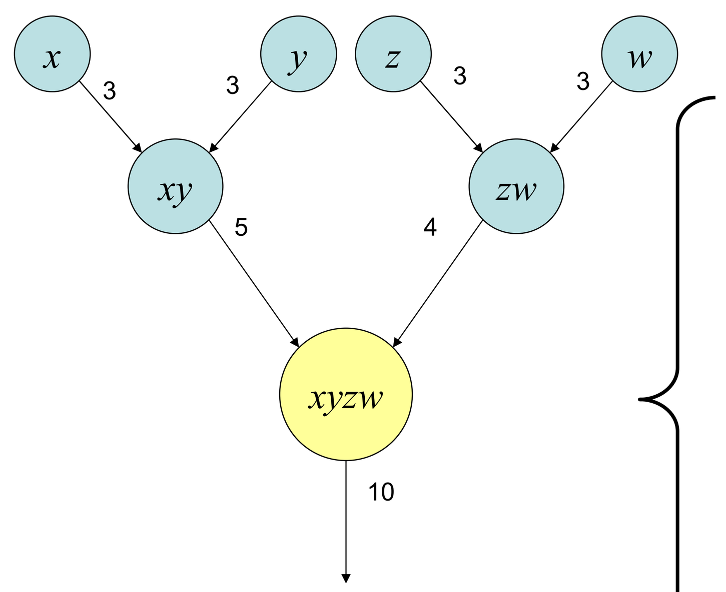
$$\begin{pmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \end{pmatrix} \begin{matrix} y=5 \\ y=-1 \\ y=-4 \end{matrix}$$

Univariate Interpolation
- Newton (Degree Based)
- Vandermonde (#Term based)

$(659-770\,y)\,x^3 + (-11 - 1092\,y^2)\,x + 54$ $z = 7, w = 11$

$x\,y^2, x^3, x^3y, x, 1$ skeleton

Zippels Algorithm Generalization



$-1092\,x\,y^2 + 659\,x^3 - 770\,x^3\,y - 11\,x + 54$ $-319\,z^2 + 60\,z\,w^2 - 378\,z^2w + 47w$

$x\,y^2, x^3, x^3y, x, 1$ skeleton z^2w, z^2, zw^2, w

l points r points

$$V = \begin{pmatrix} xy^2 & x^3 & x^3y & x & 1 \\ x^2y^4 & x^6 & x^6y^2 & x^2 & 1 \\ x^3y^6 & x^9 & x^9y^6 & x^3 & 1 \\ x^4y^8 & x^{12} & x^{12}y^8 & x^4 & 1 \\ x^5y^{10} & x^{15} & x^{15}y^{10} & x^5 & 1 \end{pmatrix}_{\substack{z=7\\w=11}} \qquad W = \begin{pmatrix} z^2w & z^2 & zw^2 & 1 \\ z^4w^2 & z^4 & z^2w^4 & 1 \\ z^6w^3 & z^6 & z^3w^6 & 1 \\ z^8w^4 & z^8 & z^4w^8 & 1 \end{pmatrix}_{\substack{x=3\\y=5}}$$

Evaluate black box at $l \times r$ points

$$E = \begin{pmatrix} f(3,5,7,11) & f(3,5,7^2,11^2) & f(3,5,7^3,11^3) & f(3,5,7^4,11^4) \\ f(3^2,5^2,7,11) & f(3^2,5^2,7^2,11^2) & f(3^2,5^2,7^3,11^3) & f(3^2,5^2,7^4,11^4) \\ f(3^3,5^3,7,11) & f(3^3,5^3,7^2,11^2) & f(3^3,5^3,7^3,11^3) & f(3^3,5^3,7^4,11^4) \\ f(3^4,5^4,7,11) & f(3^4,5^4,7^2,11^2) & f(3^4,5^4,7^3,11^3) & f(3^4,5^4,7^4,11^4) \end{pmatrix}$$

$$C = V^{-1} \times E \times (W^{-1})^T$$ Coefficient Matrix

$$C = \begin{matrix} & z^2w & z^2 & zw^2 & w \\ xy^2 & 0 & -5 & -1 & 0 \\ x^3 & 1 & 2 & 0 & 2 \\ x^3y & -3 & 0 & 1 & 0 \\ x & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 2 & -4 \end{matrix}$$

$$f(x,y,z,w) = -5\,x\,y^2z^2 - x\,y^2z\,w^2 + x^3z^2w + 2\,x^3z^2 + 2\,x^3w - 3\,x^3y\,z^2w + x^3y\,z\,w^2 - x\,w + 2\,z^2 - 4\,w$$

Variable	No	Pts	(s)Est	Evals	(s)Stage	(s)Eval	(s)Vand	(s)Intrp	Polynomial Terms	Polynomial Deg
rc	1	29	0.0	0	0.0	0.0	0.0	0.0	29	28
ra	2	25	2.4	696	2.5	2.4	0.1	0.0	287	36
rs	3	15	16.3	4018	16.7	14.0	2.6	0.2	2176	46
rb	4	11	147.2	21760	153.5	81.2	71.5	0.8	2176	46
Classic				26474	172.7					

Generalized

5

Univariate images in positions: rc=6 ra=5 rs=4 rb=3

Multivariate Image	No	(s)Est	Evals	(s)Stage	(s)Eval	(s)Poly	(s)Vand	(s)Intrp	Polynomial Terms	Polynomial Deg
5/6	2	2.5	672	2.6	2.4	0.0	0.1	0.1	287	36
3/4	1	0.5	140	0.5	0.5	0.0	0.0	0.0	107	22
1/2	0	137.6	30316	136.0	108.2	0.7	7.9	19.3	2176	46
			31128	139.7						

Computer Algebra in Scientific Computing, CASC 2005. Kalamata, Greece, Sep 12-16