

# Modified Affine Arithmetic in Tensor Form <sup>★</sup>

Huahao Shou <sup>a,b,\*</sup>, Hongwei Lin <sup>b</sup>, Ralph Martin <sup>c</sup>,  
Guojin Wang <sup>b</sup>

<sup>a</sup>*Department of Applied Mathematics, Zhejiang University of Technology,  
Hangzhou 310014, China.*

<sup>b</sup>*State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, China.*

<sup>c</sup>*School of Computer Science, Cardiff University, Cardiff, CF24 3XF, UK.*

---

## Abstract

This paper extends the *modified affine arithmetic in matrix form* method for bivariate polynomial evaluation and algebraic curve plotting in 2D to *modified affine arithmetic in tensor form* for trivariate polynomial evaluation and algebraic surface plotting in 3D. Experimental comparison shows that modified affine arithmetic in tensor form is not only more accurate but also much faster than affine arithmetic when evaluating trivariate polynomials.

*Key words:* Interval arithmetic, Affine arithmetic, Algebraic surfaces

---

## 1 Introduction

Affine arithmetic (AA) was first introduced by Comba and Stolfi [1] in 1993 as an improvement over interval arithmetic (IA). Due to its ability to keep track of correlations between variables in subexpressions, AA is often more resistant to over-conservatism when evaluating estimates of ranges of expressions over intervals. AA has been successfully applied as a replacement for IA in many geometric and computer graphics applications.

---

<sup>★</sup> Project supported jointly by the National Natural Science Foundation of China (No. 60373033 & No. 60333010), the National Natural Science Foundation for Innovative Research Groups (No.60021201) and the Foundation of State Key Basic Research 973 Item (No. 2002CB312101).

\* Corresponding author.

*Email address:* shh@zjut.edu.cn (Huahao Shou).

However AA still has an over-conservatism problem because it uses approximation when multiplying affine forms, and thus it can be further improved to give so-called modified affine arithmetic (MAA). This uses a matrix form for bivariate polynomial evaluation which keeps all noise terms without any approximation; we have used it for algebraic curve plotting in 2D [2,5]. In this paper we extend MAA to tensor form for trivariate polynomial evaluation, and illustrate its use for algebraic surface plotting in 3D. In a previous paper [4] we proved theoretically that MAA is more accurate than both interval arithmetic using centered form (IAC), and ordinary AA. Clearly, we also expect MAA in tensor form to also be more efficient than ordinary AA, and we validate this expectation with an efficiency comparison.

## 2 Modified Affine Arithmetic in Tensor Form

Let  $f(x, y, z)$  be a polynomial in three variables expressed in power form and  $\Omega$  be a box-shaped interval of interest:

$$f(x, y, z) = \sum_{i=0}^n \sum_{j=0}^m \sum_{k=0}^l A_{ijk} x^i y^j z^k, \quad (x, y, z) \in \Omega = [\underline{x}, \bar{x}] \times [\underline{y}, \bar{y}] \times [\underline{z}, \bar{z}].$$

We rewrite  $f(x, y, z)$  in tensor representation  $f(x, y, z) = X \otimes_x (Z \otimes_z A) \otimes_y Y$ , where  $X = (1, x, \dots, x^n)$ ,  $Y = (1, y, \dots, y^m)^T$ , and  $Z = (1, z, \dots, z^l)$  are vectors of powers, and  $A_{ijk}$  is the coefficient tensor.

Let us now convert the interval forms  $[\underline{x}, \bar{x}]$ ,  $[\underline{y}, \bar{y}]$  and  $[\underline{z}, \bar{z}]$  to affine forms:  $\hat{x} = x_0 + x_1 \varepsilon_x$ ,  $\hat{y} = y_0 + y_1 \varepsilon_y$ ,  $\hat{z} = z_0 + z_1 \varepsilon_z$ , where  $\varepsilon_x$ ,  $\varepsilon_y$  and  $\varepsilon_z$  are noise symbols whose values are unknown but each is assumed to be in the range  $[-1, 1]$ . Here,  $x_0 = (\bar{x} + \underline{x})/2$ ,  $y_0 = (\bar{y} + \underline{y})/2$ ,  $z_0 = (\bar{z} + \underline{z})/2$ ,  $x_1 = (\bar{x} - \underline{x})/2$ ,  $y_1 = (\bar{y} - \underline{y})/2$ , and  $z_1 = (\bar{z} - \underline{z})/2$ . We also define power vectors of the noise symbols  $\hat{X} = (1, \varepsilon_x, \dots, \varepsilon_x^n)$ ,  $\hat{Y} = (1, \varepsilon_y, \dots, \varepsilon_y^m)^T$ ,  $\hat{Z} = (1, \varepsilon_z, \dots, \varepsilon_z^l)$ . We now define three further matrices  $B$ ,  $C$  and  $D$  as follows. Firstly,

$$B = \begin{bmatrix} 1 & x_0 & \cdots & x_0^{n-1} & x_0^n \\ 0 & x_1 & \cdots & (n-1)x_0^{n-2}x_1 & nx_0^{n-1}x_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & x_1^{n-1} & nx_0x_1^{n-1} \\ 0 & 0 & \cdots & 0 & x_1^n \end{bmatrix};$$

$$\text{more completely: } B_{ij} = \begin{cases} \binom{j}{i} x_0^{j-i} x_1^i, & i \leq j \\ 0, & i > j \end{cases}, \quad i = 0, 1, \dots, n; \quad j = 0, 1, \dots, n.$$

Also,

$$C = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ y_0 & y_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ y_0^{m-1} & (m-1)y_0^{m-2}y_1 & \cdots & y_1^{m-1} & 0 \\ y_0^m & my_0^{m-1}y_1 & \cdots & my_0y_1^{m-1} & y_1^m \end{bmatrix};$$

$$\text{in detail } C_{ij} = \begin{cases} 0, & i < j \\ \binom{i}{j} y_0^{i-j} y_1^j, & i \geq j \end{cases}, \quad i = 0, 1, \dots, m; \quad j = 0, 1, \dots, m.$$

Finally

$$D = \begin{bmatrix} 1 & z_0 & \cdots & z_0^{l-1} & z_0^l \\ 0 & z_1 & \cdots & (l-1)z_0^{l-2}z_1 & lz_0^{l-1}z_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & z_1^{l-1} & lz_0z_1^{l-1} \\ 0 & 0 & \cdots & 0 & z_1^l \end{bmatrix};$$

$$\text{in detail } D_{ij} = \begin{cases} \binom{j}{i} z_0^{j-i} z_1^i, & i \leq j \\ 0, & i > j \end{cases}, \quad i = 0, 1, \dots, l; \quad j = 0, 1, \dots, l.$$

We now have that  $X = \hat{X}B$ ,  $Y = C\hat{Y}$ ,  $Z = \hat{Z}D$ . We may compute the tensor  $G$  from matrices  $B$ ,  $C$  and  $D$ , and the original coefficient tensor  $A$  as follows:  $G = B \otimes_x (D \otimes_z A) \otimes_y C$ , giving

$$f(\hat{x}, \hat{y}, \hat{z}) = \hat{X} \otimes_x (\hat{Z} \otimes_z G) \otimes_y \hat{Y} = \sum_{i=0}^n \sum_{j=0}^m \sum_{k=0}^l G_{ijk} \varepsilon_x^i \varepsilon_y^j \varepsilon_z^k.$$

Up to now the calculation is exact without any approximation. Furthermore, it can be seen that this polynomial is actually the centered form of the original polynomial [3]. A tighter interval for the range of values taken on by  $f(x, y, z)$  inside the box,  $[\underline{F}, \overline{F}]$ , compared to the interval provided by standard IA on the centered form, can be obtained as follows:

$$\begin{aligned} \overline{F} &= G_{000} + \sum_{k=1}^l \left\{ \begin{array}{l} \max(0, G_{00k}), \text{ if } k \text{ is even} \\ |G_{00k}|, \text{ otherwise} \end{array} \right\} + \\ &\quad \sum_{j=1}^m \sum_{k=0}^l \left\{ \begin{array}{l} \max(0, G_{0jk}), \text{ if } j, k \text{ are both even} \\ |G_{0jk}|, \text{ otherwise} \end{array} \right\} + \end{aligned}$$

$$\sum_{i=1}^n \sum_{j=0}^m \sum_{k=0}^l \left\{ \begin{array}{ll} \max(0, G_{ijk}), & \text{if } i, j, k \text{ are all even} \\ |G_{ijk}|, & \text{otherwise} \end{array} \right\},$$

and

$$\begin{aligned} \underline{F} = & G_{000} + \sum_{k=1}^l \left\{ \begin{array}{ll} \min(0, G_{00k}), & \text{if } k \text{ is even} \\ -|G_{00k}|, & \text{otherwise} \end{array} \right\} + \\ & \sum_{j=1}^m \sum_{k=0}^l \left\{ \begin{array}{ll} \min(0, G_{0jk}), & \text{if } j, k \text{ are both even} \\ -|G_{0jk}|, & \text{otherwise} \end{array} \right\} + \\ & \sum_{i=1}^n \sum_{j=0}^m \sum_{k=0}^l \left\{ \begin{array}{ll} \min(0, G_{ijk}), & \text{if } i, j, k \text{ are all even} \\ -|G_{ijk}|, & \text{otherwise} \end{array} \right\}. \end{aligned}$$

### 3 Comparison of AA and MAA in Tensor Form by Examples

For reasons of space we show here only one representative example of the application of MAA in tensor form: consider the surface  $(2x^2 + y^2 + z^2 - 1)^3 - 0.1x^2z^3 - y^2z^3 = 0$  inside the box  $[-1.25, 1.25] \times [-1.25, 1.25] \times [-1.25, 1.25]$ . This degree 6 polynomial equation, taken from Taubin's paper [6], represents a heart-shaped surface. Graphical outputs produced by using a recursive subdivision algorithm to plot this surface using AA, and MAA in tensor form, respectively, are shown in Figures 1 and 2. Table 1 shows that AA is clearly worse in terms of accuracy and computational effort, while MAA in tensor form is slightly better than IAC. We observed similar phenomena for other test examples.

Table 1

Comparison of AA, MAA in tensor form and IAC for heart surface

Methods	Voxels plotted	Subdivisions involved	CPU time used
AA	143104	61901	45 min 33 sec 781 msec
IAC	63168	31669	2 min 26 sec 922 msec
MAA	59104	28333	2 min 16 sec 219 msec

### 4 Conclusions

MAA in tensor form is not only more accurate but also much faster than AA. We may also theoretically demonstrate that MAA in tensor form is similar to



Fig. 1. Heart surface plotted by AA      Fig. 2. Heart surface plotted by MAA in tensor form

IAC, but enhanced by a proper consideration of the properties of even or odd powers of polynomial terms. As a result, using MAA in tensor form is always slightly more accurate than IAC, while IAC is always more accurate than AA. In conclusion we recommend that the MAA in tensor form be used instead of AA or IAC in trivariate geometric computations.

## References

- [1] J. L. D. Comba and J. Stolfi, Affine arithmetic and its applications to computer graphics, *Anais do VII SIBGRAPI* (1993), 9–18. Available at <http://www.dcc.unicamp.br/~stolfi/>.
- [2] R. Martin, H. Shou, I. Voiculescu, A. Bowyer and G. Wang, Comparison of interval methods for plotting algebraic curves, *Computer Aided Geometric Design* 19(7) (2002), 553–587.
- [3] H. Ratschek and J. Rokne, *Computer Methods for the Range of Functions*, Ellis Horwood, 1984.
- [4] H. Shou, H. Lin, R. Martin and G. Wang, Modified affine arithmetic is more accurate than centered interval arithmetic or affine arithmetic, In: Michael J. Wilson, Ralph R. Martin (Eds.), *Mathematics of Surfaces*, Springer-Verlag Berlin Heidelberg New York, *Lecture Notes in Computer Science* 2768, (2003), 355–365.
- [5] H. Shou, R. Martin, I. Voiculescu, A. Bowyer and G. Wang, Affine arithmetic in matrix form for polynomial evaluation and algebraic curve drawing, *Progress in Natural Science* 12(1) (2002), 77–80.
- [6] G. Taubin, Rasterizing algebraic curves and surfaces, *IEEE Computer Graphics Appl.* 14(2) (1994), 14–23.