A technique for DDA seed shifting and scaling

John Kerl

Feb 8, 2001

Abstract

This paper describes a simple, unified technique for DDA seed shifting and scaling. As well, the terms *DDA*, *seed*, *shifting* and *scaling* are defined.

1 Background

Let $P(x)$ be a polynomial of degree N, over the real numbers. Let $x_s, s = 0, 1, 2, \ldots$, be an evenly spaced input mesh, i.e. $x_s = x_0 + sh$ for a mesh width h. We define the first difference of $P(x_s)$ to be

$$\Delta P(x_s) = P(x_{s+1}) - P(x_s) \tag{1}$$

The second difference of $P(x_s)$ is defined as

$$\begin{aligned} \Delta^2 P(x_s) &= \Delta P(x_{s+1}) - \Delta P(x_s) && (2) \\ &= P(x_{s+2}) - 2P(x_{s+1}) + P(x_s) && (3) \end{aligned}$$

The zeroth difference is defined as $\Delta^0 P(x_s) = P(x_s)$, and higher-order differences are defined as $\Delta^{j+1} P(x_s) = \Delta^j P(x_{s+1}) - \Delta^j P(x_s)$.

The $N + 1$ differences $c_j = \Delta^j P(x_0)$ are called *DDA seeds* (*DDA* is defined below) for $P(x)$ on the input mesh x_s. The first $N + 1$ polynomial outputs

$P(x_0), P(x_1), \ldots, P(x_N)$ may be represented as a vector $\mathbf{D}$, as may the seeds $\mathbf{c}$. The two are related by the following matrix $\mathbf{A}$, with rows i and columns j (using zero-based indices):

$$\begin{aligned} \mathbf{A}_{ij} &= \binom{i}{j} && (4) \\ \mathbf{A}^{-1}_{ij} &= (-1)^{i+j}\binom{i}{j} && (5) \\ \mathbf{c} &= \mathbf{A}^{-1}\mathbf{D} && (6) \\ \mathbf{D} &= \mathbf{A}\mathbf{c} && (7) \end{aligned}$$

with the definition

$$\binom{i}{j} = \begin{cases} \frac{i!}{j!(i-j)!}, & i \geq j; \\ 0, & i < j \end{cases} \quad (8)$$

The matrix $\mathbf{A}^{-1}$ extracts finite differences of $P(x_0)$, using $P(x_0)$ through $P(x_N)$; the matrix $\mathbf{A}$ turns the DDA seeds back into the first $N+1$ polynomial outputs on the mesh. (Note that both $\mathbf{A}$ and $\mathbf{A}^{-1}$ are lower-triangular. The non-zero entries of $\mathbf{A}$ are simply the contents of the Pascal triangle; the non-zero entries of $\mathbf{A}^{-1}$ are the same, but with alternating signs.)

The polynomial $P(x)$ is completely specified by the input mesh x_s along with any one of the following: the $N+1$ polynomial coefficients, the $N+1$ outputs on the mesh, or the $N+1$ DDA seeds for the mesh. In engineering practice the latter is often convenient since it allows one to reproduce the polynomial outputs, for many more mesh points than just the first $N+1$, using only addition, as follows.

A digital differential analyzer (*DDA*) is defined by the following system of recurrence relations:

Initial conditions	Update rules
$R_0(0) = c_0$	$R_0(s+1) = R_0(s) + R_1(s)$
$R_1(0) = c_1$	$R_1(s+1) = R_1(s) + R_2(s)$
$R_2(0) = c_2$	$R_2(s+1) = R_2(s) + R_3(s)$
...	...
$R_{N-1}(0) = c_{N-1}$	$R_{N-1}(s+1) = R_{N-1}(s) + R_N(s)$
$R_N(0) = c_N$	$R_N(s+1) = R_N(s)$

The register $R_0(s)$ is the output of the DDA, and represents $P(x_s)$. The remaining registers, $R_1(s)$ through $R_N(s)$, preserve the internal state of the DDA from one step to the next. The initial values of the registers are simply the seeds $\mathbf{c}$ calculated using finite differences of $P(x_0)$, as described above.

Since only $R_0(s)$ is of interest, we often rename the output of the DDA at time step s simply $D(s)$. Then $D(s)$ is a sequence, the values of which equal $P(x_s)$ on the mesh x_s.

Advantages of using DDAs for polynomial evaluation include the following: (1) Since only addition is used at each step, a DDA allows efficient evaluation of polynomials by circumventing the need for floating-point multiplies. (2) The N additions at each time step may be done in parallel. (3) Further efficiency may be obtained if the addition is done in fixed-point. In fact, fixed-point DDA hardware circuitry can evaluate a polynomial of arbitrary order at the rate of one sample per processor clock cycle.

Disadvantages of using DDAs include the following: (1) Given a set of seeds, one may obtain values of $P(x)$ only on the specified mesh. (This paper describes a technique for modifying seeds to generate values on a different mesh). (2) Values must be obtained sequentially, e.g. obtaining $P(37)$ requires having first computed $P(36)$, etc. (3) It can be shown that round-off error accumulates as s increases, and is approximately equal to the product of the initial seed error and the Nth power of the step number.

2 Problem statement

It is often necessary to modify DDA seeds to generate output for a different mesh than originally specified. If the original mesh is x_0+sh, then modifying the seeds to produce output over the new mesh $(x_0+\delta h)+sh$ is called *shifting*.

Modifying the seeds to produce output over the new mesh $x_0 + \alpha sh$ is called *scaling*. Various ad hoc methods for shifting or scaling, particularly for non-negative integer δ or α, have been presented in my paper *The ABCs of DDAs* and elsewhere. This paper describes the general resampling problem for changing the mesh from $x_0 + sh$ to $(x_0 + \delta h) + \alpha sh$, yielding a single technique for shifting and scaling. This common formula is equivalent to the previous ad-hoc methods, as special cases.

3 Technique

We transform the $N+1$ original seeds into the first $N+1$ polynomial outputs. We then find the Lagrange interpolating polynomial for $P(x)$ given the input mesh $x_s = x_0 + sh$ and the $N + 1$ outputs $P(x_0 + sh)$, and evaluate this polynomial on the new mesh $x_0 + \delta h + \alpha sh$ to obtain the $N + 1$ new outputs $P(x_0 + \delta h + \alpha sh)$. Lastly, we transform the first $N + 1$ polynomial outputs on the new mesh back to $N + 1$ new DDA seeds.

In general, given polynomial inputs $x_0, x_1, \ldots, x_N$ and outputs $y_0, y_1, \ldots, y_N$, the Lagrange interpolator $L(x)$ is

$$L(x) = \sum_{t=0}^{N} \prod_{k=0}^{N,k\neq t} y_t \frac{x - x_k}{x_t - x_k} \tag{9}$$

Since our y_t will be the $N + 1$ outputs of our Nth-degree polynomial $P(x)$ evaluated on the original mesh, and since the Lagrange interpolator is unique, $L(x)$ is exactly $P(x)$.

Equation 9 becomes:

$$\begin{aligned} P(x_0 + \delta h + \alpha sh) &= \sum_{t=0}^{N} \prod_{k=0}^{N,k\neq t} P(x_0 + th) \frac{(x_0 + \delta h + \alpha sh) - (x_0 + kh)}{(x_0 + th) - (x_0 + kh)} & (10) \\ &= \sum_{t=0}^{N} \prod_{k=0}^{N,k\neq t} P(x_0 + th) \frac{(\delta + \alpha s - k)}{(t - k)} & (11) \end{aligned}$$

Equivalently, expressing equation 10 using matrix notation, the $N+1$-element

vector $D'(s) = P(x_0 + \delta h + \alpha s h), s = 0, 1, \ldots, N$, is related to the $N+1$-element vector $D(t) = P(x_0 + th), t = 0, 1, \ldots, N$, by the following matrix $\mathbf{Q}(\delta, \alpha)$ with rows s and columns t:

$$Q_{st}(\delta, \alpha) = \prod_{k=0}^{N, k \neq t} \frac{\delta + \alpha s - k}{t - k} \tag{12}$$

Thus

$$\mathbf{c}' = \mathbf{A}^{-1}\mathbf{Q}(\delta, \alpha)\mathbf{A}\mathbf{c} \tag{13}$$

4 Examples

Here are tabulated $\mathbf{A}^{-1}\mathbf{Q}(\delta, \alpha)\mathbf{A}$ for various values of δ and α, in the cubic case.

$\delta = 0, \alpha = 1$ (identity transformation):

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\delta = 0, \alpha = 2$ (scale only; modified seeds produce even-numbered samples of original sequence):

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 8 \end{bmatrix}$$

$\delta = 1, \alpha = 2$ (shift and scale; modified seeds produce odd-numbered samples of original sequence):

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 2 & 3 & 1 \\ 0 & 0 & 4 & 8 \\ 0 & 0 & 0 & 8 \end{bmatrix}$$

$\delta = 0, \alpha = 3$ (scale only; modified seeds produce every third output sample):

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 3 & 3 & 1 \\ 0 & 0 & 9 & 18 \\ 0 & 0 & 0 & 27 \end{bmatrix}$$

$\delta = 0, \alpha = 1/2$ (scale only; modified seeds produce output twice as finely):

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1/2 & -1/8 & 1/16 \\ 0 & 0 & 1/4 & -1/8 \\ 0 & 0 & 0 & 1/8 \end{bmatrix}$$

$\delta = 1, \alpha = 1$ (shift only; modified seeds produce output starting at x_1):

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\delta = 2, \alpha = 1$ (shift only; modified seeds produce output starting at x_2):

$$\begin{bmatrix} 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\delta = -1, \alpha = 1$ (shift only; modified seeds produce output starting at x_{-1}):

$$\begin{bmatrix} 1 & -1 & 1 & -1 \\ 0 & 1 & -1 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\delta = 1.1, \alpha = 0.837$ (shift and scale):

$$\begin{bmatrix} 1 & 1.1000 & 0.0550 & -0.0165 \\ 0 & 0.8370 & 0.8525 & -0.0026 \\ 0 & 0 & 0.7006 & 0.6564 \\ 0 & 0 & 0 & 0.5864 \end{bmatrix}$$