Coefficient Path Calculation

The F-Node Formula in PolyPaint

PolyPaint Technical Documentation

Abstract

This document gives the exact formula for the **final coefficient vector** $\mathbf{F}(t) \in \mathbb{C}^n$ — the complex polynomial coefficients fed to the Ehrlich–Aberth solver at each time step. Every root trajectory the application produces is determined entirely by this vector. A degree-d polynomial has $n = d + 1$ coefficients indexed $i = 0, \ldots, n-1$.

Master Formula

$$\boxed{F_i(t) \;=\; M\big(\,C_i(t),\; D_i(t),\; \theta(t)\big) \;+\; J_i(s) \;\cdot\; \mathbf{1}_{[i \in S]}} \tag{1}$$

Symbol	Meaning
$F_i(t) \in \mathbb{C}$	Final coefficient i at time t. This is what the solver sees.
$C_i(t) \in \mathbb{C}$	C-node i position at time t, sampled from its precomputed curve.
$D_i(t) \in \mathbb{C}$	D-node i position at time t, sampled from its precomputed curve.
$M(c, d, \theta)$	Morph interpolation function (identity when morph disabled).
$\theta(t)$	Morph phase angle $= 2\pi \cdot r_{\text{morph}} \cdot t$.
$J_i(s) \in \mathbb{C}$	Jiggle offset for coefficient i at jiggle step s (additive).
s	Jiggle step $= \lfloor t/\Delta t_{\text{jiggle}} \rfloor$.
$S \subseteq \{0, \ldots, n-1\}$	Set of selected coefficient indices (jiggle targets).
$\mathbf{1}_{[i \in S]}$	Iverson bracket: 1 if $i \in S$, 0 otherwise.

The order of operations is:

$$\text{animate } C \;\longrightarrow\; \text{animate } D \;\longrightarrow\; \text{morph blend} \;\longrightarrow\; \text{add jiggle} \;\longrightarrow\; \text{solve}$$

Jiggle is never applied to C or D individually—it is applied to the already-blended result. When morph is disabled, M is the identity on C: $M(c, d, \theta) = c$. When jiggle is off, the jiggle term is zero.

$C_i(t)$: C-Node Animation

Each coefficient i has a **home position** $\mathrm{H}_i = \gamma_i[0] \in \mathbb{C}$ and a precomputed closed curve γ_i consisting of N sample points in the complex plane. The curve is computed once from parameters $(\mathrm{H}_i, \text{pathType}, R, \alpha, \text{extra})$.

Curve Sampling

Define the **phase** of coefficient i at time t:

$$\varphi_i(t) \;=\; t \cdot v_i \cdot \delta_i \qquad \text{where} \quad \delta_i = \begin{cases} -1 & \text{if CCW} \\ +1 & \text{if CW} \end{cases}$$

and v_i is the speed parameter. Then the normalized phase:

$$u_i \;=\; \big((\varphi_i \bmod 1) + 1\big) \bmod 1 \;\in\; [0, 1)$$

and the raw index into the curve array:

$$\rho \;=\; u_i \cdot N$$

Smooth curves. Linear interpolation between adjacent samples (wrapping at N):

$$\ell = \lfloor \rho \rfloor \bmod N, \qquad h = (\ell + 1) \bmod N, \qquad f = \rho - \lfloor \rho \rfloor \tag{2}$$

$$C_i(t) = \gamma_i[\ell]\,(1 - f) \;+\; \gamma_i[h]\,f \tag{3}$$

Cloud curves (random path, `_isCloud` flag): no interpolation—snap to the nearest point:

$$C_i(t) = \gamma_i\big[\lfloor \rho \rfloor \bmod N\big]$$

Dither. If the curve carries a dither parameter $\sigma_{\%} > 0$:

$$C_i(t) \;\leftarrow\; C_i(t) \;+\; \sigma_{\text{abs}}\,(\mathcal{N}(0,1) + i\,\mathcal{N}(0,1)), \qquad \sigma_{\text{abs}} = \frac{\sigma_{\%}}{100} \cdot E$$

where $\mathcal{N}(0,1)$ is a standard Gaussian (Box–Muller) and $E =$ `coeffExtent` is the maximum pairwise distance between any two coefficients.

Static case. When pathType = `"none"`: $C_i(t) = \mathrm{H}_i$ for all t.

Curve resolution. $N = 200$ for basic paths; $N = 1500$ for space-filling paths (Hilbert, Peano, Sierpiński, spiral).

$D_i(t)$: D-Node Animation

D-nodes use the **identical curve-sampling formula** as C-nodes, with their own independent pathType, speed, radius, angle, direction, and precomputed curve.

Follow C. When D_i.pathType = `"follow-c"`:

$$D_i(t) = C_i(t)$$

The D-node copies the already-animated C-node position. This is evaluated *after* C-node advancement.

Static case. When pathType = `"none"`: $D_i(t)$ stays at the D-node's home position (which may differ from H_i if the user has dragged it).

$M(c, d, \theta)$: Morph Interpolation

When morph is **disabled**: $M(c, d, \theta) = c$ (D-nodes are ignored).

When morph is **enabled**, the behaviour depends on the **C–D path type**. All paths share a local coordinate frame.

Setup

Given $c, d \in \mathbb{C}$ (the animated C- and D-node positions):

$$\boldsymbol{\delta} = d - c \qquad L = |\boldsymbol{\delta}| \tag{4}$$

$$\mathbf{u} = \boldsymbol{\delta}/L \qquad \mathbf{v} = i\,\mathbf{u} \quad (90° \text{ CCW rotation}) \tag{5}$$

$$\mathbf{m} = \tfrac{1}{2}(c + d) \qquad a = L/2 \quad (\text{semi-major}) \tag{6}$$

Let $\sigma = +1$ if CCW, -1 if CW. When $L < 10^{-15}$, all paths return c.

The morph phase angle:

$$\theta(t) = 2\pi \cdot r_{\text{morph}} \cdot t \tag{7}$$

where r_{morph} is the morph rate in Hz (default 0.01, range $[0, 0.1]$).

Line (default)

$$\mu = \frac{1 - \cos\theta}{2} \qquad \Longrightarrow \qquad M(c, d, \theta) = c\,(1 - \mu) + d\,\mu \tag{8}$$

Raised-cosine oscillation. μ sweeps smoothly from 0 (at c) to 1 (at d) and back. Period $= 1/r_{\text{morph}}$ seconds. The path is a straight line segment.

Circle

The c–d segment is the **diameter** of a circle:

$$M = \mathbf{m} \;+\; (-a\cos\theta)\,\mathbf{u} \;+\; (\sigma\, a \sin\theta)\,\mathbf{v} \tag{9}$$

At $\theta = 0$: position $= c$. At $\theta = \pi$: position $= d$. Full revolution $= 2\pi$.

Ellipse

Same as circle, with a shorter semi-minor axis $b = p \cdot a$ where $p \in [0.1, 1.0]$ is the minor-axis percentage:

$$M = \mathbf{m} \;+\; (-a\cos\theta)\,\mathbf{u} \;+\; (\sigma\, b \sin\theta)\,\mathbf{v} \tag{10}$$

When $p = 1$, this reduces to the circle. The major axis lies along $c \to d$.

Figure-8

A Lissajous 1:2 curve crossing at the midpoint:

$$M = \mathbf{m} \;+\; (-a\cos\theta)\,\mathbf{u} \;+\; (\sigma\, \tfrac{a}{2} \sin 2\theta)\,\mathbf{v} \tag{11}$$

At $\theta = 0$: at c. At $\theta = \pi/2$: crosses midpoint. At $\theta = \pi$: at d. At $\theta = 3\pi/2$: crosses midpoint again. The two lobes are symmetric about $\mathbf{m}$.

Summary Table

Path	**Local-frame displacement** (l_x, l_y)	**Period**
Line	$c + (d-c)\,\dfrac{1-\cos\theta}{2}$	2π
Circle	$(-a\cos\theta,\ \sigma\, a\sin\theta)$	2π
Ellipse	$(-a\cos\theta,\ \sigma\, b\sin\theta)$	2π
Figure-8	$\left(-a\cos\theta,\ \sigma\,\frac{a}{2}\sin 2\theta\right)$	2π

Position in global frame: $M = \mathbf{m} + l_x\,\mathbf{u} + l_y\,\mathbf{v}$ (except Line, which is computed directly).

$J_i(s)$: Jiggle Offsets

Jiggle produces per-coefficient **additive** complex offsets with these properties:

- Computed from **home positions** $\mathrm{H}_i = \gamma_i[0]$, *not* from the current animated position $C_i(t)$.
- **Piecewise-constant** in time: offsets change only at step boundaries, every Δt_{jiggle} seconds.
- Applied **only to selected** coefficients ($i \in S$).
- Applied **after** morph blending.

$$s = \lfloor t \,/\, \Delta t_{\text{jiggle}} \rfloor, \qquad J_i(s) = 0 \ \text{ if } \ i \notin S \text{ or mode} = \texttt{"none"}$$

Notation: $\mathrm{H}_i =$ home position; $\mathbf{c} = \frac{1}{|S|}\sum_{i\in S}\mathrm{H}_i$ (centroid of selected homes); $E = \texttt{coeffExtent}$; $\sigma = (\texttt{jiggleSigma}/10)\cdot E$; R_α denotes rotation by angle α.

Random

$$J_i(s) = \sigma\,\mathcal{N}_1(0,1) + i\,\sigma\,\mathcal{N}_2(0,1) \tag{12}$$

Independent Gaussian noise, freshly sampled each step.

Walk

$$J_i(s) = J_i(s{-}1) + \sigma\,\mathcal{N}_1(0,1) + i\,\sigma\,\mathcal{N}_2(0,1), \qquad J_i(0) = 0 \tag{13}$$

Cumulative random walk. The *only* jiggle mode with memory.

Rotate

$$\alpha = \frac{2\pi s}{N_{\text{angle}}}, \qquad J_i(s) = R_\alpha\,(\mathrm{H}_i - \mathbf{c}) - (\mathrm{H}_i - \mathbf{c}) \tag{14}$$

Rigid rotation of the selected coefficients about their centroid.

Circle (jiggle)

$$\alpha = \frac{2\pi s}{N_{\text{circle}}}, \qquad J_i(s) = R_\alpha\,\mathrm{H}_i - \mathrm{H}_i \tag{15}$$

Rotation of each coefficient about the origin.

Scale–Center

$$g = \frac{\texttt{scaleStep}}{100} \cdot E, \qquad J_i(s) = \frac{\mathrm{H}_i}{|\mathrm{H}_i|} \cdot g \cdot s \tag{16}$$

Radial growth from the origin. Linear in step number.

Scale–Centroid

$$\mathbf{d}_i = \mathrm{H}_i - \mathbf{c}, \qquad J_i(s) = \frac{\mathbf{d}_i}{|\mathbf{d}_i|} \cdot g \cdot s \tag{17}$$

Spiral–Centroid

$$\alpha = \frac{2\pi s}{N_{\text{angle}}}, \qquad r = 1 + \frac{g\,s}{|\mathbf{d}_i|}, \qquad J_i(s) = r\,R_\alpha\,\mathbf{d}_i - \mathbf{d}_i \tag{18}$$

Combined rotation and radial scaling from centroid.

Spiral–Center

$$r = 1 + \frac{g\,s}{|\mathrm{H}_i|}, \qquad J_i(s) = r\,R_\alpha\,\mathrm{H}_i - \mathrm{H}_i \tag{19}$$

Breathe

$$\lambda = 1 + \frac{A}{100}\,\sin\!\Big(\frac{2\pi s}{P}\Big), \qquad J_i(s) = (\mathrm{H}_i - \mathbf{c})\,(\lambda - 1) \tag{20}$$

Sinusoidal radial pulsation from centroid. A = amplitude parameter, P = period in steps.

Wobble

$$\alpha = \frac{2\pi}{N_{\text{angle}}}\,\sin\!\Big(\frac{2\pi s}{P}\Big), \qquad J_i(s) = R_\alpha\,(\mathrm{H}_i - \mathbf{c}) - (\mathrm{H}_i - \mathbf{c}) \tag{21}$$

Sinusoidal angular oscillation about centroid.

Lissajous

$$A_{\text{abs}} = \frac{A}{100} \cdot E, \qquad J_i(s) = A_{\text{abs}}\,\sin\!\Big(\frac{2\pi f_x s}{P} + \frac{\pi}{2}\Big) + i\,A_{\text{abs}}\,\sin\!\Big(\frac{2\pi f_y s}{P}\Big) \tag{22}$$

Uniform translation of *all* selected coefficients along a Lissajous curve. Same offset for every coefficient.

Complete Pipeline

1. **Animate C-nodes.** For each coefficient i:

$$C_i(t) = \begin{cases} \text{sample } \gamma_i \text{ at phase } \varphi_i(t) & \text{if pathType} \neq \texttt{"none"} \\ \mathrm{H}_i & \text{otherwise} \end{cases}$$

   If dither is enabled, add Gaussian noise.

2. **Animate D-nodes.** For each D-node i:

$$D_i(t) = \begin{cases} C_i(t) & \text{if pathType} = \texttt{"follow-c"} \\ \text{sample D-curve}_i \text{ at phase } \varphi_i^D(t) & \text{if pathType} \neq \texttt{"none"} \\ D_{\mathrm{H},i} & \text{otherwise} \end{cases}$$

3. **Morph blend.**

$$B_i = \begin{cases} M(C_i(t),\, D_i(t),\, \theta(t)) & \text{if morph enabled} \\ C_i(t) & \text{otherwise} \end{cases}$$

4. **Jiggle.** Compute step $s = \lfloor t/\Delta t_{\text{jiggle}} \rfloor$ and offsets $J_i(s)$ for $i \in S$.

5. **Assemble F.**

$$F_i(t) = B_i + J_i(s) \cdot \mathbf{1}_{[i \in S]}$$

6. **Solve.** Feed $\mathbf{F}(t)$ to the Ehrlich–Aberth root finder $\rightarrow$ roots.

Key Invariants

1. **Jiggle never mutates C or D.** It only affects the final blend F.
2. **Jiggle uses home positions.** All jiggle offsets (except walk) are deterministic functions of (s, H_i, params), not the current animated position.
3. **Morph uses live positions.** The morph function receives the already-animated $C_i(t)$ and $D_i(t)$, not home positions.
4. **No feedback.** $F_i(t)$ depends only on the current state of $C_i(t)$, $D_i(t)$, and jiggle step s. It does not depend on F at any previous time (except jiggle walk mode, which accumulates).
5. **In-place mutation.** The `coefficients[i].re/im` fields hold the current animated $C_i(t)$ after step 1. `solveRoots()` copies before modifying.

Worker (Fast Mode) Differences

The same pipeline runs inside Web Workers with these differences:

- **Jiggle is static per pass.** The main thread computes $J_i(s)$ once and serializes it as a flat array. Workers apply the same offset for every step within a pass. At jiggle interval boundaries, the main thread recomputes and reinitializes.
- **Non-line morph falls back to JS.** When the C–D path type $\neq$ `"line"` and the WASM step loop would otherwise be used, the code falls back to the JavaScript step loop.
- **Elapsed time per step.** $t_{\text{step}} = t_{\text{offset}} + (\text{step}/\text{totalSteps}) \cdot \Delta T_{\text{pass}}$ where ΔT_{pass} is the simulated time per pass.

The mathematical result is identical to the main-thread path.