Errata, precisions

Section 3.2.1

- Observation 3.8 / Equation (3.11), p. 28: The elements in max/min of $\tilde{l}_i$ must be taken with the absolute value.

Section 3.2.2

- The subroutine computing angles must handle the case where subsequent control points are equal.
- For the simplified angle method it is not necessary to perform the complete affine transformation, as $-x|x|$ has the desired variations, but the complete transformation is in better sync with the syntax of the rest of the program.
- Default threshold values: The threshold value using the simplified method merits a few comments. Say we will have $\alpha_\tau \in [0, 10)$ measured in degrees. The simplified method returns angles

  $$\tilde{\alpha} = -\frac{\pi}{2}\cos\alpha|\cos\alpha| + \frac{\pi}{2}. \tag{1}$$

  A regression fit between α and $\tilde{\alpha}$ on the interval specified gives an excellent ($r^2 = 1.000$) fit for a quadratic polynomial $f(x) = \tau_c x^2$, with $\tau_c = 1.543$. Se Figure 1. The correspondence is shown in red for angles $\alpha \in [0, 10)$ yielding angles $\tilde{\alpha} \in [0, 2.8)$. The regression fit is shown in black.

Section 3.2.3

- Baseline-method: default threshold value r is obtained from $\sin\frac{\pi-\alpha_{tau}}{2} = rl/l, \alpha_{tau} = \frac{\pi}{180}$ (1 degree).
- Length-ratio method: threshold value must be linked to worst-case scenario: right angle (or flat angle, beyond the point?). Approximation: $R = 1 + \delta/l_{max}$.
- The length-ratio method offers some reuse of length-calculations from the length-method. The combination of these two methods merits more study, i.e., the application of the length-method in a prephase and a second phase of refinement using the length-ratio method.

Section 3.3

- Knot placement equation: From Böhms equation, setting $\omega_i = \frac{z-t_i}{t_{i+p}-t_i}$, we have

  $$\bar{t}_i^* = \omega_i t_i^* + (1-\omega_i)t_{i-1}^*. \tag{2}$$

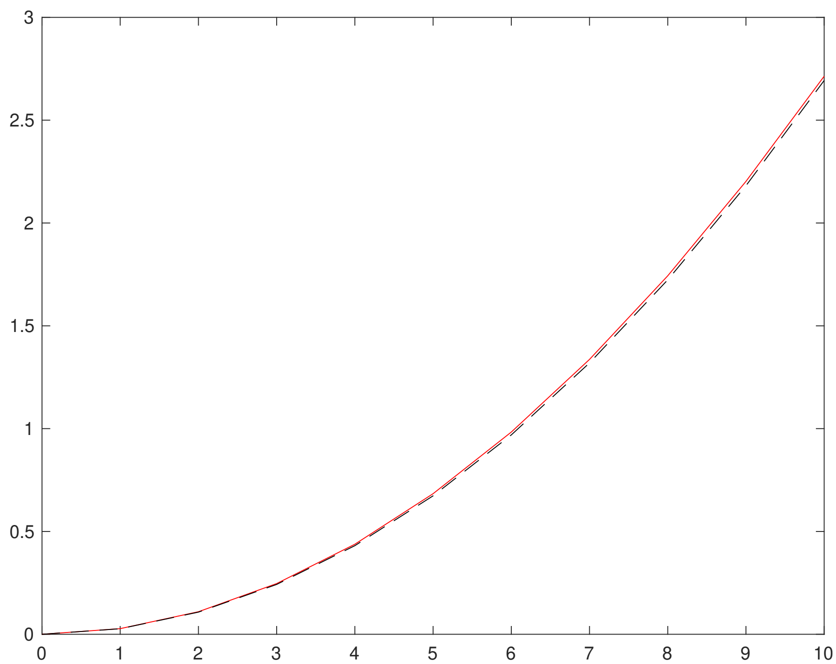

Figure 1: *Correspondence between the actual angle (x-axis) and the angle returned by the simplified method (y-axis) for small angles (degrees). The actual relationship in red, the regression fit in dashed black.*

If we want a new control point at a ratio $\omega_\mu/(1-\omega_\mu)$ between control points c_μ and $c_{\mu-1}$, then solving for z gives

$$z = \omega_\mu(t_{\mu+p} - t_\mu) + t_\mu. \tag{3}$$

This means, if we want a 'tighter' control point at c_μ, choose $\omega = 1$, i.e., set $z = t_{\mu+p}$. If we want a new control point midway between two old control points c_μ and $c_{\mu-1}$, choose $\omega = 1/2$, i.e., set $z = t_\mu + \frac{1}{2}(t_{\mu+p} - t_\mu)$.

- Knot placement: if we want to imitate Chaikin's algorithm, it can only be done one segment at a time. We cannot in general impose that the insertion of a single knot cuts all p segments in half.

- Notation 3.2: It is more clever to index while referencing 'to the left' in the control polygon, i.e., that the segment $[c_{\mu-1}, c_\mu)$ is associated with c_μ, to keep indexing closer to the indexing used in Böhms equation. We should have introduced the backwards difference $\Delta P_i = P_i - P_{i-1}$ to ease implementation of the methods.

Section 3.4

- Stopping criterion: is it cheaper/better to check the p new control points using the distance threshold ϵ, or after all better to define absolute stopping criterions depending on C? Both stopping criteria, as well as the imposed maximum number of knots to insert are implemented. The adaptive plotting function returns a flag to show which criterion caused the function to return.

- A partitioning Σ can be kept or updated between iterations. If it refers to indices, the update is straigthforward. If it refers to original control point values, then a rule must be given for how to decide the new break point indices when some of these old break point values disappear.

Cosmetics

- It is better to number figures according to chapter.

Next steps

- Algorithm: general structure, implementing more of the auxiliary functions.

- Generalize and show feasibility of methods for coefficient dimension s=3.

- Find an adequate set of default values and possibly a useful combination of methods.

- Implement extensions to tensor product surfaces.
- Implement using parallel programming. Implement in C language.
- Create package, advocate for general degree splines being geometric primitives.