

Curvature and torsion estimators based on parametric curve fitting

THOMAS LEWINER^{1,2}, JOÃO D. GOMES JR.¹, HÉLIO LOPES¹ AND MARCOS CRAIZER¹

¹ Department of Mathematics — Pontifícia Universidade Católica — Rio de Janeiro — Brazil

² Géométrie Project — INRIA – Sophia Antipolis — France
{tomlew, jgomes, lopes, craizer}@mat.puc--rio.br.

Abstract. Many applications of geometry processing and computer vision rely on geometric properties of curves, particularly their curvature. Several methods have already been proposed to estimate the curvature of a planar curve, most of them for curves in digital spaces. This work proposes a new scheme for estimating curvature and torsion of planar and spatial curves, based on weighted least-squares fitting and local arc-length approximation. The method is simple enough to admit a convergence analysis that take into account the effect of noise in the samples. The implementation of the method is compared to other curvature estimation methods showing a good performance. Applications to prediction in geometry compression are presented both as a practical application and as a validation of this new scheme.

Keywords: *Differential Geometry. Curvature Estimation. Weighted Least-Squares. Geometry Processing. Geometry Compression and Predictors.*

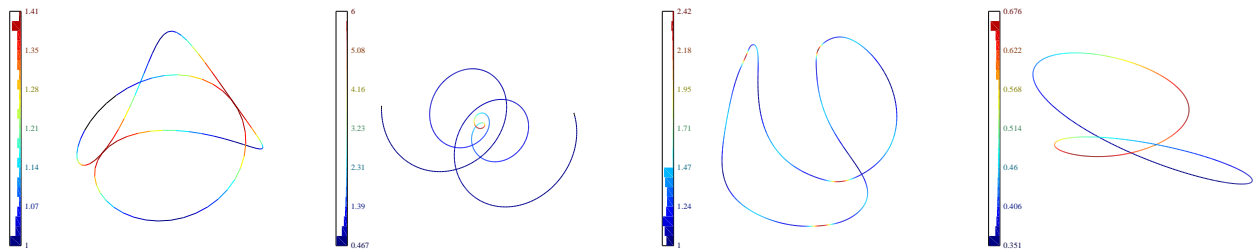


Figure 1: Spatial curves used for testing: Clelia, Conical Helix, Baseball and Toric Solenoid. The color indicates curvature.

1 Introduction

Many applications of geometry processing [19] and computer vision [18] rely on geometric properties of curves. Two of the most fundamental characteristics of a curve are its curvature and torsion, which measure how a curve bends. Curvature motion [22], curve reconstruction [1, 3], adaptive curve approximation [17] and geometry compression [13] are examples of contemporary computer graphics applications that require accurate curvature estimation.

Motivation. Several methods have already been proposed for curvature estimation, most of them for the particular case of digital spaces, i.e. curves extracted from images [8]. This work studies sampled curves, i.e. piecewise-linear approximations of a smooth curve. This general framework,

which includes the digital curves, permits a clear theoretical analysis. Moreover, it is well suited for applications to geometric modelling and computer graphics. In the context of these two areas, sampled curves may proceed from different sources: polygonal approximation of parametric or implicit curves, curve acquisition, curve reconstruction, among others.

The original motivation of this work was to develop an accurate curvature and torsion estimator to serve as predictor for geometric encoding, particularly for implicit curve compression schemes [13] and triangular mesh compression schemes, such as the Edgebreaker [21, 14]. In both situations the predictor must guess the position of the next vertex to be encoded based on the sequence of points that have already been transmitted.

Problem statement. A piecewise linear approximation of a curve \mathbf{r} is an ordered finite collection of points $\{\mathbf{p}_i\}$ of the curve, called samples. This sampled curve may contain some noise, i.e., the points \mathbf{p}_i stay close to the curve, but

Preprint MAT. 16/05, communicated on March 1st, 2005 to the Department of Mathematics, Pontifícia Universidade Católica — Rio de Janeiro, Brazil. The corresponding work was published in Computers & Graphics, volume 29, number 5, pp. 641–655. Elsevier, october 2005.

not necessarily lie on it. This paper presents estimators for the tangent vector and the curvature of a curve \mathbf{r} at a sample point \mathbf{p} of $\{\mathbf{p}_i\}$. For the case of spatial curves, it also proposes a torsion estimator.

Contributions. This paper introduces a new method for curvature and torsion estimation based on weighted least-squares fitting. More precisely, the method approximate the samples by a parametric curve that have one of its coordinate functions given by a second (or third)-order polynomial. The paper shows the convergence of the estimators under reasonable conditions over the sampling of the curve and the amplitude of the noise. It also provides a practical implementation of this method. The results show that the proposed method compares nicely to the state-of-the-art, and that it has a strong stability over different conditions of noise and sampling.

Paper outline. Section 2 *Curvature and torsion of a parametric curve* introduces the concepts and notations from differential geometry of curves that will be used along this work. Section 3 *Previous and related works* discusses the previous and related works. Section 4 *Theoretical framework* presents the theoretical analysis of the new method. Section 5 *Computational framework* gives details of the implementation of the scheme. The algorithm is compared to the state-of-the-art in section 6 *Experimental results*. Finally, section 7 *Applications to compression* introduces an application of the proposed curvature estimator to geometry compression.

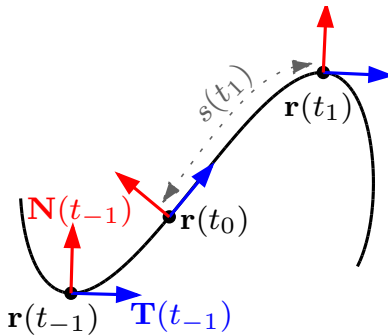


Figure 2: The arc-length $s(t)$ helps defining the tangent and the normal vectors $\mathbf{T}(t)$ and $\mathbf{N}(t)$.

2 Curvature and torsion of a parametric curve

A *parametric curve* is a smooth function $\mathbf{r} : I \subset \mathbb{R} \rightarrow \mathbb{R}^n$. When $n = 2$ it is called a planar curve and when $n = 3$ it is called spatial curve. The parametric curve \mathbf{r} is said to be *regular* if $\mathbf{r}'(t) = \frac{d\mathbf{r}}{dt}(t)$ never vanishes on I . From now on, \mathbf{r} will denote a regular parameterized curve.

The *arc-length* s from the point $\mathbf{r}(t_0)$, $t_0 \in I$, to a given point $\mathbf{r}(t_1)$, $t_1 \in I$, is defined by $s(t_1) = \int_{t_0}^{t_1} \|\mathbf{r}'(u)\| du$. When the curve is regular, $s(t)$ is strictly increasing, and has therefore an inverse $t(s)$. The curve can be *parameterized by the arc-length* by considering $\mathbf{r}(s) = \mathbf{r} \circ t(s)$.

The vector $\mathbf{T}(s) = \mathbf{r}'(s)$ is called the *tangent vector*. In the case of planar curves, the *normal vector* $\mathbf{N}(s)$ is obtained from the tangent vector by a rotation of 90 degrees in the anti-clockwise sense. The vectors $\mathbf{T}(s)$ and $\mathbf{N}(s)$ are collinear, i.e. there exists a function $\kappa(s)$ such that $\mathbf{T}'(s) = \kappa(s)\mathbf{N}(s)$, which is called the *curvature* of the curve at the point $\mathbf{r}(s)$.

The curvature at a point also corresponds to the variation of the tangent direction with respect to the arc-length: $\kappa(s) = \theta'(s)$, where $\theta(s) = \angle(\mathbf{T}(s), (1, 0))$ is the angle of the tangent vector with a fixed direction. The sign of $\kappa(s)$ indicates whether the curve is locally convex or concave at point $\mathbf{r}(s)$. The curvature $k(s)$ is also the inverse of the radius of the osculating circle at $\mathbf{r}(s)$. When the curve $\mathbf{r}(t)$ is not parameterized by the arc-length, its curvature is given by

$$\kappa(t) = \frac{\mathbf{r}' \times \mathbf{r}''}{\|\mathbf{r}'\|^3} \quad (1)$$

For spatial curve, the tangent vector is again defined by $\mathbf{T}(s) = \mathbf{r}'(s)$. The normal vector is defined by $\mathbf{N}(s) = \mathbf{r}''(s)/\|\mathbf{r}''(s)\|$, and the *bi-normal vector* is given by the cross-product of $\mathbf{T}(s)$ and $\mathbf{N}(s)$, i.e., $\mathbf{B}(s) = \mathbf{T}(s) \times \mathbf{N}(s)$.

For spatial curve, the curvature $\kappa(s)$, also defined by the formula $\mathbf{T}'(s) = \kappa(s)\mathbf{N}(s)$, is always positive. The *torsion* is defined by the formula $\mathbf{B}'(s) = \tau(s)\mathbf{N}(s)$. When the curve $\mathbf{r}(t)$ is not parameterized by the arc-length, its curvature is given by the absolute value of equation ((1)) and the torsion is given by

$$\tau(t) = -\frac{(\mathbf{r}' \times \mathbf{r}'') \cdot \mathbf{r}'''}{\|\mathbf{r}' \times \mathbf{r}''\|^2}. \quad (2)$$

For more details related to curvature and torsion, see [4].

3 Previous and related works

Several methods have already been proposed for estimating curvature, most of them for planar curves. This section introduces the most significant ones. All these methods have been implemented for the comparisons of section 6 *Experimental results*. They are classified in three groups: the methods that use gaussian smoothing, the methods that estimate curvature directly from three points and the methods that use least squares approximations.

(a) Methods based on gaussian smoothing

The methods considered in this section were developed for digital curves, but they can be easily adapted to sampled curves. The first method of [25] computes the curvature at a point \mathbf{p}_0 from the convolution of the estimated angle $\hat{\theta}$ with

the derivative of the gaussian kernel (G_σ), as follows:

$$\hat{\kappa} = \hat{\theta} * G'_\sigma, \quad \text{with } \hat{\theta}(j) = \tan^{-1} \left(\frac{y_{j+1} - y_j}{x_{j+1} - x_j} \right).$$

where

$$G'_\sigma(j) = -2\sigma j e^{-\sigma j^2}.$$

A variation of this method is obtained by making a re-sampling using linear interpolation and then applying the method. This variation can be written as

$$\hat{\kappa} = \frac{\hat{\theta} * G'_\sigma}{1.107}, \quad \text{with } \hat{\theta}(j) = \tan^{-1} \left(\frac{y_{j+1}^r - y_j^r}{x_{j+1}^r - x_j^r} \right),$$

where (x^r, y^r) are the new samples and the constant 1.107 is the re-sampling bias in the digital curve context [25].

The last method of [25] uses equation (1) to estimate the first and second derivative directly by the convolution of the coordinate functions with the derivatives of the gaussian kernel:

$$\hat{\mathbf{r}}' = \mathbf{r} * G'_\sigma, \quad \hat{\mathbf{r}}'' = \mathbf{r} * G''_\sigma.$$

The last method of this group, described in [9], is based on the Fast Fourier Transform. In the time domain, this method also corresponds to a convolution with the gaussian kernel. For this method, one considers the samples $\mathbf{r}(n) = (x(n), y(n))$, $n = 0, \dots, N-1$, as a complex signal $\mathbf{u}(n) = x(n) + iy(n)$, where $i^2 = -1$. A basic property of the discrete Fourier transform is that the derivative of a signal $u(n)$ corresponds to the multiplication by i of the transform $U(s)$. Based on this property, the first and second derivatives of $u(n)$ are estimated in the transform domain and then the inverse discrete Fourier transform is applied.

In the comparisons of section 6, these methods will be respectively referred as G1, G2, G3 and G4.

(b) Methods that estimates curvature from three points

The first method of this group, proposed in [6], is based on the angle variation in a neighborhood of the point. Considering the points $(\mathbf{p}_{-q}, \mathbf{p}_0, \mathbf{p}_q)$, the estimated curvature at \mathbf{p}_0 is given by

$$\hat{\kappa}(\mathbf{p}_0) = \frac{\angle(\mathbf{p}_{-q}\mathbf{p}_0, \mathbf{p}_0\mathbf{p}_q)}{\|\mathbf{p}_{-q}\mathbf{p}_0\| + \|\mathbf{p}_0\mathbf{p}_q\|},$$

where $\angle(\mathbf{p}_{-q}\mathbf{p}_0, \mathbf{p}_0\mathbf{p}_q)$ is the angle between the vectors $\mathbf{p}_0\mathbf{p}_{-q}$ and $\mathbf{p}_0\mathbf{p}_q$.

A variation of this method, proposed in [11], estimates the curvature from the external angle. The estimated curvature at \mathbf{p}_0 is given by $\hat{\kappa}(\mathbf{p}_0) = \phi_0 / \left(2l_0 \cos\left(\frac{\phi_0}{2.1}\right) \right)$, where ϕ_0 is the external angle $(\pi - \angle(\mathbf{p}_{-q}\mathbf{p}_0, \mathbf{p}_0\mathbf{p}_q))$ and $l_0 = \frac{\|\mathbf{p}_{-q}\mathbf{p}_0\| + \|\mathbf{p}_0\mathbf{p}_q\|}{2}$.

Another method, presented in [7], is based on an approximation of the osculating circle at a point \mathbf{p}_0 . It approximates it by the circle circumscribing the triangle $(\mathbf{p}_{-q}, \mathbf{p}_0, \mathbf{p}_q)$. The radius R of this circle is given by

$$R = \frac{\|\mathbf{p}_{-q}\mathbf{p}_0\| \cdot \|\mathbf{p}_0\mathbf{p}_q\| \cdot \|\mathbf{p}_{-q}\mathbf{p}_q\|}{4 \cdot \text{area}(\mathbf{p}_{-q}, \mathbf{p}_0, \mathbf{p}_q)}.$$

The last method of this group, cited in [2], uses the same three points $(\mathbf{p}_{-q}, \mathbf{p}_0, \mathbf{p}_q)$ to estimate the first and second derivatives at \mathbf{p}_0 . These derivatives are computed by

$$r' = \frac{\mathbf{p}_q - \mathbf{p}_0}{\|\mathbf{p}_0\mathbf{p}_q\|} + \frac{\mathbf{p}_0 - \mathbf{p}_{-q}}{\|\mathbf{p}_{-q}\mathbf{p}_0\|} - \frac{\mathbf{p}_q - \mathbf{p}_{-q}}{\|\mathbf{p}_{-q}\mathbf{p}_0\| + \|\mathbf{p}_0\mathbf{p}_q\|}$$

$$r'' = 2 \cdot \frac{(\mathbf{p}_{-q} - \mathbf{p}_0) \cdot \|\mathbf{p}_0\mathbf{p}_q\| + (\mathbf{p}_q - \mathbf{p}_0) \cdot \|\mathbf{p}_{-q}\mathbf{p}_0\|}{\|\mathbf{p}_{-q}\mathbf{p}_0\| \cdot \|\mathbf{p}_0\mathbf{p}_q\| \cdot (\|\mathbf{p}_{-q}\mathbf{p}_0\| + \|\mathbf{p}_0\mathbf{p}_q\|)}.$$

The methods of this group will be respectively referred as T1, T2, T3 and T4.

(c) Methods that use least squares approximation

The methods presented in this section estimate the curvature from a window of $2q + 1$ points around p_0 , with $q \geq 1$. They use least square approximation and have the advantage of reducing the effect of noise in the sampling.

Circle fitting

This method fits a circle to the sample points of the window, using [20], and deduces the curvature from the inverse of the radius of the fitted circle. The algorithm of [20] finds the coefficients of the circle equation $A(x^2 + y^2) + Bx + Cy + D = 0$ by least squares approximation. In this algorithm, the value of A was fixed as 1. Figure 3 illustrates one circle obtained by this method.

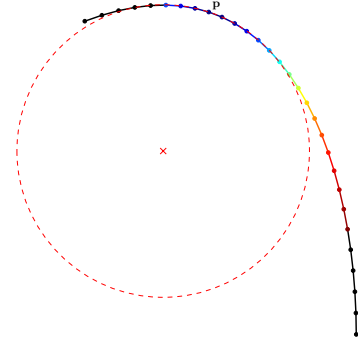


Figure 3: Osculating circle estimation at a point p of the curve $r(t) = (\sin(t), \sin(t) \cdot \cos(t)) : t \in [\frac{\pi}{5}, \frac{\pi}{2}]$ sampled with 30 points and using a centered window with 5 points.

As mentioned in [20], the restriction $A = 1$ generates instability for circles of big radius. This phenomenon can be observed on Figure 4. If another coefficient is fixed, the problem for low curvature can be solved, but appears for other classes of curves. The analysis of the impact of noise on the curvature approximation is studied in [24].

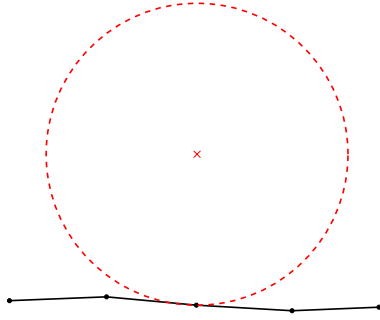


Figure 4: Circle fitting at a point of very low curvature.

Implicit parabola fitting

In this method, proposed in [5], the curve is described as a graph of $y = f(x)$ or $x = f(y)$. The variation of x and y inside the window determines which parameterization to use : if the variation of x is bigger than the one of y , the algorithm takes $y = f(x)$, and vice-versa. As both cases are similar, only the case $y = f(x)$ will be described next.

In order to simplify the notation, assume that $\mathbf{p}_0 = (0, 0)$. The objective is to find f'_0 and f''_0 that minimize

$$E_x(f'_0, f''_0) = \sum_{i=-q}^q (y_i - f'_0 x_i - \frac{1}{2} f''_0 x_i^2)^2 .$$

The solution of this least squares problem is given by

$$f'_0 = \frac{cg - bh}{ac - b^2} , \quad f''_0 = \frac{ah - bg}{ac - b^2} ,$$

where

$$\begin{cases} a = \sum_{i=-q}^q x_i^2 , & g = \sum_{i=-q}^q x_i y_i , \\ b = \frac{1}{2} \sum_{i=-q}^q x_i^3 , & h = \frac{1}{2} \sum_{i=-q}^q x_i^2 y_i , \\ c = \frac{1}{4} \sum_{i=-q}^q x_i^4 . \end{cases} .$$

With these estimates, the curvature is then easily computed. This method is a particular case of [5], where the approximating polynomial is limited to the second order.

4 Theoretical framework

This section introduces a new approach for the problem of curvature and torsion estimation.

(a) Model and notations

Consider a collection of samples $\{p_i\}$ of a planar or a spacial smooth curve \mathbf{r} , i.e., a finite sequence of *sample points* of \mathbf{r} , eventually perturbed by a random noise. In this theoretical analysis, the curve is assumed to be parameterized by the arc-length. The curvature estimation for planar cuves needs an approximation of the first and second derivatives of $\mathbf{r}(s)$, while for torsion estimation also needs an approximation of the third derivatives of $\mathbf{r}(s)$.

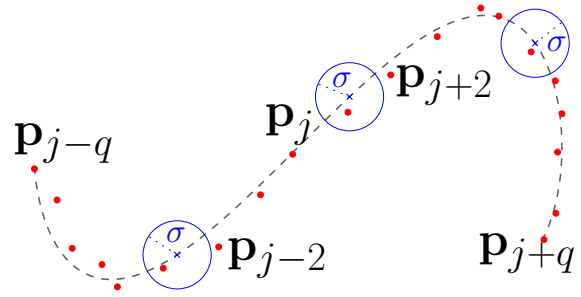


Figure 6: Sampled curve with noise.

Consider a fixed sample point \mathbf{p}_0 . The estimation of the derivatives of r at \mathbf{p}_0 will be performed from a window of $2q + 1$ points around \mathbf{p}_0 : $\{\mathbf{p}_{-q}, \mathbf{p}_{-q+1}, \dots, \mathbf{p}_q\}$.

The noise at a point \mathbf{p}_i is modelled by a random vector η_i , normal to r at \mathbf{p}_i , and the random variables η_i are assumed to be independent and identically distributed (i.i.d.), with zero mean and variance σ^2 .

Considering that $\mathbf{p}_0 = \mathbf{r}(0)$ is the origin, the second order approximation can be written as :

$$\mathbf{r}(s) = \mathbf{r}'(0) s + \frac{1}{2} \mathbf{r}''(0) s^2 + \mathbf{g}(s) s^3$$

the third order approximation as :

$$\mathbf{r}(s) = \mathbf{r}'(0) s + \frac{1}{2} \mathbf{r}''(0) s^2 + \frac{1}{6} \mathbf{r}'''(0) s^3 + \mathbf{g}(s) s^4 ,$$

with $\mathbf{g}(s) \rightarrow 0$ when $s \rightarrow 0$. Let s_i be the arc-length corresponding to the sample \mathbf{p}_i . The second order approximation can now be written

$$\mathbf{p}_i = \mathbf{r}'(0) s_i + \frac{1}{2} \mathbf{r}''(0) s_i^2 + \mathbf{g}(s_i) s_i^3 + \eta_i$$

and the third order also :

$$\mathbf{p}_i = \mathbf{r}'(0) s_i + \frac{1}{2} \mathbf{r}''(0) s_i^2 + \frac{1}{6} \mathbf{r}'''(0) s_i^3 + \mathbf{g}(s_i) s_i^4 + \eta_i$$

(b) The weighted least squares approach

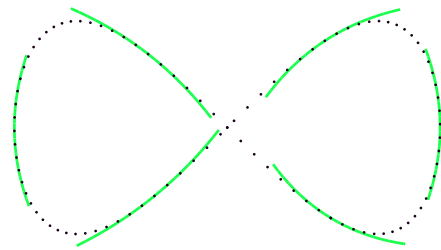


Figure 7: Second-order weighted least square fitting.

The estimates of $\mathbf{r}'(0)$, $\mathbf{r}''(0)$ and $\mathbf{r}'''(0)$ are obtained by a weighted least squares minimisation. The weight w_i of point \mathbf{p}_i must be positive, relatively large for small $|s_i|$ and relatively small for large $|s_i|$. For example, one can consider

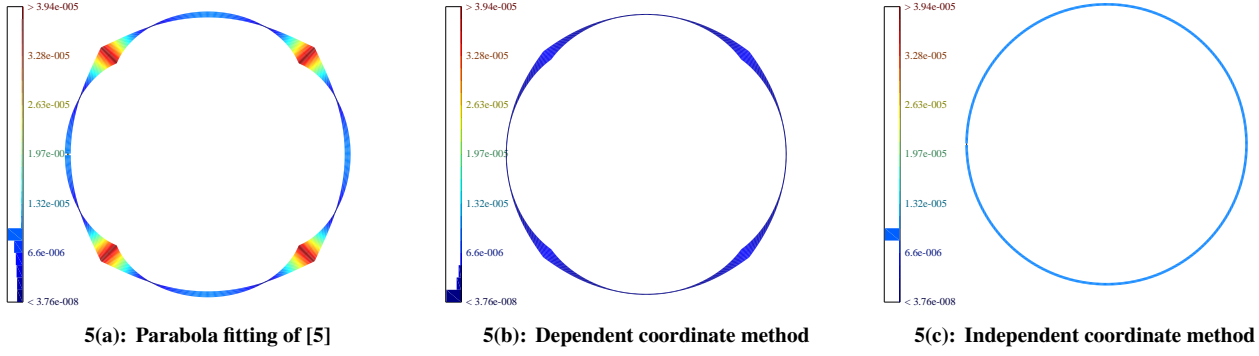


Figure 5: The independent coordinate method is rotation invariant: big errors were drawn by wide line, the scale being the same on the three figures.

weights of the form $w_i = \alpha \exp(-\beta s_i^2)/s_i^k$, or simply $w_i = 1$.

The arc-length s_i can be estimated as follows: let Δl_k be the length of the vector $\mathbf{p}_k \mathbf{p}_{k+1}$, where k ranges from $-q$ to $(q-1)$. Then, the arc-length from \mathbf{p}_0 to \mathbf{p}_i can be approximated by $l_i = \sum_{k=0}^{i-1} \Delta l_k$, when $i > 0$, and $l_i = -\sum_{k=i}^{-1} \Delta l_k$, when $i < 0$.

(c) Curvature Estimators for planar curves

Dependent coordinates method. Consider the case of a planar curve. The idea of the dependent coordinate method is to locally fit a parametric curve $(\hat{x}(s), \hat{y}(s))$ to the curve, with one of the coordinate functions, say \hat{x} , being quadratic in the arc-length: $\hat{x}(s) = x_0 + x'_0 \cdot s + \frac{1}{2} x''_0 \cdot s^2$. This coordinate will be called the *independent* coordinate, as opposed to the *dependent* coordinate \hat{y} whose derivatives will be deduced from those of \hat{x} . The dependent coordinate is chosen according to a simple criterion, for example the angle of the tangent vector, minimal variance or the χ^2 test.

The derivatives x'_0 and x''_0 of \hat{x} can be estimated by minimizing :

$$E_x(x'_0, x''_0) = \sum_{i=-q}^q w_i (x_i - x'_0 l_i - \frac{1}{2} x''_0 (l_i)^2)^2. \quad (3)$$

The estimates y'_0 and y''_0 for the dependent coordinate's derivatives $y'(0)$ and $y''(0)$ are obtained by both the unit norm of the tangent and the orthogonality of the tangent and the normal:

$$\begin{cases} (x'_0)^2 + (y'_0)^2 = 1 \\ x'_0 x''_0 + y'_0 y''_0 = 0 \end{cases}. \quad (4)$$

The minimization of equation ((3)) can be written in terms of matrix inversion [12], which leads to a direct resolution (see section 5(a) *Dependent coordinates method for planar curves* for the implementation):

$$\begin{bmatrix} a_1 & a_2 \\ a_2 & a_3 \end{bmatrix} \cdot \begin{bmatrix} x'_0 \\ x''_0 \end{bmatrix} = \begin{bmatrix} b_{x,1} \\ b_{x,2} \end{bmatrix}, \quad (5)$$

where

$$\begin{cases} a_1 = \sum_{i=-q}^q w_i l_i^2, & b_{x,1} = \sum_{i=-q}^q w_i l_i x_i \\ a_2 = \frac{1}{2} \sum_{i=-q}^q w_i l_i^3, & b_{x,2} = \frac{1}{2} \sum_{i=-q}^q w_i l_i^2 x_i \\ a_3 = \frac{1}{4} \sum_{i=-q}^q w_i l_i^4 \end{cases}$$

Independent coordinates method. A variation of the above method is to estimate the derivatives of the dependent coordinate function \hat{y} , in the same way as the derivatives of the independent coordinate \hat{x} : the estimations of y'_0 and y''_0 are obtained from the following matrix equation :

$$\begin{bmatrix} a_1 & a_2 \\ a_2 & a_3 \end{bmatrix} \cdot \begin{bmatrix} y'_0 \\ y''_0 \end{bmatrix} = \begin{bmatrix} b_{y,1} \\ b_{y,2} \end{bmatrix}, \quad (6)$$

$$\text{where } \begin{cases} b_{y,1} = \sum_{i=-q}^q w_i l_i (y_i) \\ b_{y,2} = \frac{1}{2} \sum_{i=-q}^q w_i (l_i)^2 (y_i) \end{cases}.$$

The tangent vector \mathbf{T} is obtained by normalizing the vector $\mathbf{r}'_0 = (x'_0, y'_0)$, while the normal vector is obtained by applying a rotation of 90 degrees on \mathbf{T} .

Whereas both coordinates cannot be exactly quadratic in the arc-length, the approximation of the independent coordinates method has the advantage of being rotation invariant (see Figure 5).

(d) Curvature estimator for spatial curves

In the case of spatial curves, a generalization of the independent coordinates method for curvature estimation is obtained as follows: The estimators x'_0 , x''_0 , y'_0 and y''_0 are given by formulas ((5)) and ((6)) respectively. And the estimators z'_0 and z''_0 for the third coordinate derivatives $z'(0)$ and $z''(0)$ are given in a similar way by the matrix equation

$$\begin{bmatrix} a_1 & a_2 \\ a_2 & a_3 \end{bmatrix} \cdot \begin{bmatrix} z'_0 \\ z''_0 \end{bmatrix} = \begin{bmatrix} b_{z,1} \\ b_{z,2} \end{bmatrix}, \quad (7)$$

$$\text{where } \begin{cases} b_{z,1} = \sum_{i=-q}^q w_i l_i (z_i) \\ b_{z,2} = \frac{1}{2} \sum_{i=-q}^q w_i (l_i)^2 (z_i) \end{cases}.$$

Equation ((1)) then gives the curvature using the above first and second order derivatives estimates.

(e) Torsion estimator for spatial curves

For spatial curves, the torsion estimator fits a cubic parametric curve to the sample points. Assuming that $\mathbf{p}_0 = \mathbf{r}(0) = (0, 0, 0)$, x'_0, x''_0 and x'''_0 should minimize

$$E_x(x'_0, x''_0, x'''_0) = \sum_{i=-q}^q w_i \left(x_i - (x'_0 s_i + \frac{1}{2} x''_0 s_i^2 + \frac{1}{6} x'''_0 s_i^3) \right)^2.$$

Considering again l_i as an approximation of s_i , the above equation can be solved by matrix inversion :

$$\begin{bmatrix} a_1 & a_2 & a_4 \\ a_2 & a_3 & a_5 \\ a_4 & a_5 & a_6 \end{bmatrix} \cdot \begin{bmatrix} x'_0 \\ x''_0 \\ x'''_0 \end{bmatrix} = \begin{bmatrix} b_{x,1} \\ b_{x,2} \\ b_{x,3} \end{bmatrix},$$

where $\begin{cases} a_4 = \frac{1}{6} \sum_{i=-q}^q w_i (l_i)^4 \\ a_5 = \frac{1}{12} \sum_{i=-q}^q w_i (l_i)^5 \\ a_6 = \frac{1}{36} \sum_{i=-q}^q w_i (l_i)^6 \\ b_{x,3} = \frac{1}{6} \sum_{i=-q}^q w_i l_i^3(x_i) \end{cases}$.

A similar approach is used to compute $y'_0, y''_0, y'''_0, z'_0, z''_0$ and z'''_0 . Using the above derivatives estimates, equation (2) gives the torsion estimate.

(f) Convergence analysis of curvature estimators

This section analyses the proposed algorithm in term of their convergence. The first part considers sampled curve without noise while the second part considers sampled curve with noise. It is important to mention that there are proofs of convergence only for curvature estimators, not for the torsion estimator. Nevertheless, the torsion estimator had experimentally shown a typical convergence behavior (see Figure 9(b) and Figure 10(d)).

Sampled curve without noise

Consider the following notation:

$$\begin{aligned} \delta &= \max\{|l_{-q}|, |l_q|\} \\ K_0 &= \max\{|\kappa(s)|, -\delta \leq s \leq \delta\}, \\ K_1 &= \max\{|\kappa'(s)|, -\delta \leq s \leq \delta\}, \\ T_0 &= \max\{|\tau(s)|, -\delta \leq s \leq \delta\}, \end{aligned}$$

where $\kappa(s)$ is the curvature and $\tau(s)$ the torsion of \mathbf{r} at s .

The curve regularity and sampling conditions will be formulated as follow. The product $K_0\delta$ should be small, which corresponds to dense sampling in high curvature regions. If not, some samples are too far from \mathbf{p}_0 to be correctly used in the estimate of the first derivatives of \mathbf{r} at \mathbf{p}_0 . Then, the products $K_1\delta$ and $T_0\delta$ should also be small, which corresponds to dense sampling in regions where curvature is changing rapidly. If this does not occur, some samples must be considered too far from the basic point to help in the second derivatives estimates.

In order to make precise these statements, let

$$\begin{aligned} \phi_1 &= \frac{l_i}{4L} \left(\sum_{i=-q}^q w_i l_i \sum_{i=-q}^q w_i l_i^4 - \sum_{i=-q}^q w_i l_i^2 \sum_{i=-q}^q w_i l_i^3 \right) \\ \phi_2 &= \frac{l_i^2}{4L} \left(\sum_{i=-q}^q w_i l_i^2 \sum_{i=-q}^q w_i l_i^2 - \sum_{i=-q}^q w_i l_i \sum_{i=-q}^q w_i l_i^3 \right). \end{aligned}$$

Denote also $\text{sinc}(\varepsilon) = \frac{\sin(\varepsilon)}{\varepsilon}$, $L = a_1 a_3 - a_2^2$ and:

$$d_1(\varepsilon) = \frac{x'(0)(1 - \text{sinc}(\varepsilon/2))}{\text{sinc}(\varepsilon/2)} + \frac{\varepsilon}{\text{sinc}(\varepsilon/2)}$$

$$d_2(\varepsilon) = x''(0) \frac{1 - \text{sinc}(\varepsilon/2)^2}{\text{sinc}(\varepsilon/2)^2} + \frac{\varepsilon}{\text{sinc}(\varepsilon/2)^2} \left(1 + \frac{x'(0)K_0}{6} \right).$$

Denote by \mathbf{r}'_0 and \mathbf{r}''_0 the estimations obtained for the coordinate derivatives by using the dependent or the independent coordinate method. The following proposition is proved in [15]:

Proposition 1 (a) Assume that $K_0\delta \leq \varepsilon$. Then

$$|\mathbf{r}'_0 - \mathbf{r}'(0)| \leq \phi_1 \left(d_1(\varepsilon) + \left| \frac{x''(0)\delta}{2} \right| \right).$$

(b) Assume that $K_0\delta \leq \varepsilon$, $K_1\delta \leq \varepsilon$ and $T_0\delta \leq \varepsilon$. Then

$$|\mathbf{r}''_0 - \mathbf{r}''(0)| \leq \phi_2 d_2(\varepsilon).$$

Observe that ϕ_1 and ϕ_2 are homogeneous of degree zero, i.e., they do not change if all l_i are multiplied by some constant. The above proposition says that the curvature estimation is convergent in the sense that reducing sufficiently the value of δ without changing the proportions of the l_i , the difference between the real and estimated will be arbitrarily small.

Sampled curve with noise

This study assumes that the noise at each sample is a random vector η_i , orthogonal to the curve at $\mathbf{r}(s_i)$, independent for each sample, with mean 0 and standard deviation σ . In order to have a good estimate of the derivatives, one must assume that the noise is not too big relatively to the distance between samples.

Denote by $\delta_1 = \min(|l_1|, |l_{-1}|)$. If someone wants to use the above estimations in the noisy case, the ratios $\frac{\sigma}{\delta_1}$ and $\frac{\sigma}{\delta_1^2}$ should be small. If not, the noise is too strong to guarantee the estimation for $\mathbf{r}'(0)$ and $\mathbf{r}''(0)$, respectively. The following proposition is proved in [15]:

Proposition 2 (a) Assume that $\sigma \leq \gamma\delta_1$. Then the error of estimation $|\mathbf{r}'_0 - \mathbf{r}'(0)|$ is bounded by the sum of the errors

of proposition 1(a) and a random variable of zero mean and variance less than $\psi_1\gamma$, where

$$\frac{4L}{\delta_1} \psi_1^2 = \sum_{i=-q}^q w_i^2 l_i^2 \left(\sum_{i=-q}^q w_i l_i^4 \right)^2 + \sum_{i=-q}^q w_i^2 l_i^4 \left(\sum_{i=-q}^q w_i l_i^3 \right)^2$$

(b) Assume that $\sigma \leq \gamma\delta_1^2$. Then the error of estimation $|\mathbf{r}''_0 - \mathbf{r}''(0)|$ is bounded by the sum of the errors of proposition 1(b) and a random variable of zero mean and variance less than $\psi_2\gamma$, where

$$\frac{2L}{\delta_1^2} \psi_2^2 = \sum_{i=-q}^q w_i^2 l_i^4 \left(\sum_{i=-q}^q w_i l_i^2 \right)^2 + \sum_{i=-q}^q w_i^2 l_i^2 \left(\sum_{i=-q}^q w_i l_i^3 \right)^2$$

Observe that ψ_1 and ψ_2 also are homogeneous of degree zero. The above proposition says that the curvature estimation is convergent in the sense that if reducing sufficiently the noise standard deviation σ without changing the proportions of the l_i , the difference between the real and estimated will be arbitrarily small.

In the particular case where the samples are symmetrically distributed around \mathbf{p}_0 and the weights w_i are equal to 1, then

$$\frac{4}{\delta_1} \psi_1^2 = \left(\sum_{i=-q}^q l_i^2 \right)^{-1} \quad \text{and} \quad \frac{2}{\delta_1^2} \psi_2^2 = \left(\sum_{i=-q}^q l_i^4 \right)^{-1}.$$

If q is big, $\psi_1 = O(q^{-3/2})$ and $\psi_2 = O(q^{-5/2})$.

5 Computational framework

The methods introduced in this work are extremely simple to implement. The implementations follow directly from the description of section 4 *Theoretical framework*.

(a) Dependent coordinates method for planar curves

Before solving the weighted least squares problem, Algorithm 1 computes the coefficients $a_1, a_2, a_3, b_{x,1}, b_{x,2}, b_{y,1}$ and $b_{y,2}$. Then algorithm 2 solves the weighted least squares problem for the independent coordinate, and deduces the estimations of the dependent coordinate. The choice of which coordinate is dependent is done considering the angle of the estimated tangent: if $|x'_0| < |y'_0|$, the x will be the independent coordinate. This algorithm also estimates the tangent and normal vectors.

(b) Independent coordinates method for planar curves

Algorithm 3 implements the independent coordinates method. It computes first x'_0, x''_0, y'_0 and y''_0 and then it estimates the curvature, the tangent and normal vectors.

Algorithm 1 Set weighted least squares variables

```

1:  $\mathbf{l}[] = a_1 = a_2 = a_3 = b_{x,1} = b_{x,2} = b_{y,1} = b_{y,2} = 0$ ;
2: for  $i = -q \dots q$  do
3:    $\mathbf{l}[i] \leftarrow \mathbf{l}[i-1] + \|p_{i-1} p_i\|$ ;
4: end for
5:  $m = \mathbf{l}[0]$ ;
6: for  $i = -q \dots q$  do
7:    $\mathbf{l}[i] \leftarrow \mathbf{l}[i] - m$ ; // Centering  $l$  on 0
8:    $w = \text{weight}(\mathbf{l}[i])^2$ ;
9:    $a_1 \leftarrow a_1 + w (\mathbf{l}[i])^2$ ;
10:   $a_2 \leftarrow a_2 + \frac{w}{2} (\mathbf{l}[i])^3$ ;
11:   $a_3 \leftarrow a_3 + \frac{w}{4} (\mathbf{l}[i])^4$ ;
12:   $b_{x,1} \leftarrow b_{x,1} + w (\mathbf{l}[i]) (x_i - x_0)$ ;
13:   $b_{y,1} \leftarrow b_{y,1} + w (\mathbf{l}[i]) (y_i - y_0)$ ;
14:   $b_{x,2} \leftarrow b_{x,2} + \frac{w}{2} (\mathbf{l}[i])^2 (x_i - x_0)$ ;
15:   $b_{y,2} \leftarrow b_{y,2} + \frac{w}{2} (\mathbf{l}[i])^2 (y_i - y_0)$ ;
16: end for
17:  $d = a_1 a_3 - a_2^2$ ; // determinant

```

Algorithm 2 Dependent coordinates for planar curves

```

1: call Set Weighted Least Squares Variables;
2:  $x'_0 = (a_3 b_{x,1} - a_2 b_{x,2})/d$ ;
3:  $y'_0 = (a_3 b_{y,1} - a_2 b_{y,2})/d$ ;
4: if  $|x'_0| < |y'_0|$  then //  $x$  is the independent coordinate
5:    $y'_0 = \text{sign}(y'_0) \sqrt{1 - \mathbf{T}_x^2}$ ;
6:    $x''_0 = (a_1 b_{x,2} - a_2 b_{x,1})/d$ ;
7:    $y''_0 = -(x'_0 y'_0)/\mathbf{T}_y$ ;
8: else //  $y$  is the independent coordinate
9:    $x'_0 = \text{sign}(x'_0) \sqrt{1 - \mathbf{T}_y^2}$ ;
10:   $y''_0 = (a_1 b_{y,2} - a_2 b_{y,1})/d$ ;
11:   $x''_0 = -(y'_0 y''_0)/\mathbf{T}_x$ ;
12: end if
13:  $\kappa = x'_0 y''_0 - y'_0 x''_0$ ;
14:  $\mathbf{T} = (x'_0, y'_0)$ ;
15:  $\mathbf{N} = \text{sign}(\kappa) (-\mathbf{T}_y, \mathbf{T}_x)$ ;

```

Algorithm 3 Independent coordinates method for planar curves

```

1: call Set Weighted Least Squares Variables;
2:  $x'_0 = (a_3 b_{x,1} - a_2 b_{x,2})/d$ ;
3:  $y'_0 = (a_3 b_{y,1} - a_2 b_{y,2})/d$ ;
4:  $x''_0 = (a_1 b_{x,2} - a_2 b_{x,1})/d$ ;
5:  $y''_0 = (a_1 b_{y,2} - a_2 b_{y,1})/d$ ;
6:  $\kappa = (x'_0 y''_0 - y'_0 x''_0) / \|(x'_0, y'_0)\|^3$ ;
7:  $\mathbf{T} = (x'_0, y'_0) / \|(x'_0, y'_0)\|$ ;
8:  $\mathbf{N} = \text{sign}(\kappa) (-\mathbf{T}_y, \mathbf{T}_x)$ ;

```

(c) Curvature estimator for space curves

The first step to implement the generalization of the algorithm 4 to estimate the curvature of spatial curves using

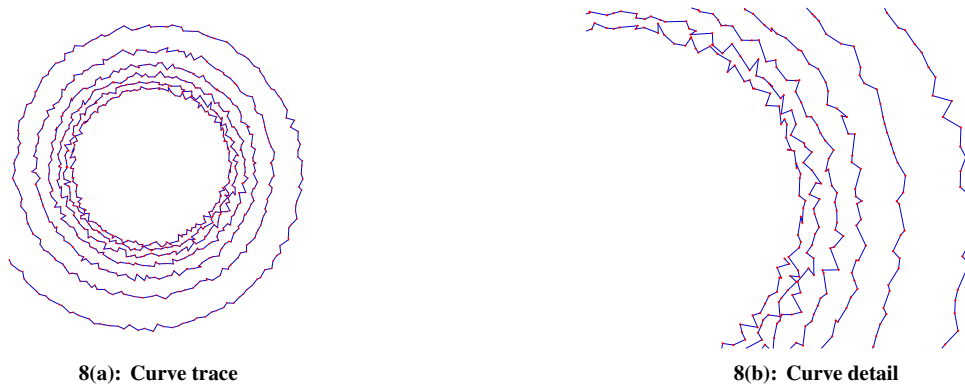


Figure 8: Noisy spiral curve (1000 samples, $\sigma = 1$).

independent coordinates is to modify the routine of algorithm 1 in order to compute $b_{z,1}$ and $b_{z,2}$. Algorithm 4 details the implementation of this method. This algorithm provides an estimation for the curvature, and also for the tangent, normal and binormal vectors. The tangent is on the direction of \mathbf{r}' . The normal is obtained by applying a Gram-Schmidt orthogonalization process to the vectors \mathbf{T} and \mathbf{r}'' . Finally, the binormal vector is a cross product of \mathbf{T} and \mathbf{N} .

Algorithm 4 Independent coordinates method for space curves

- 1: call Set Weighted Least Squares Variables;
 - 2: $x'_0 = (a_3 b_{x,1} - a_2 b_{x,2})/d$;
 - 3: $y'_0 = (a_3 b_{y,1} - a_2 b_{y,2})/d$;
 - 4: $z'_0 = (a_3 b_{z,1} - a_2 b_{z,2})/d$;
 - 5: $x''_0 = (a_1 b_{x,2} - a_2 b_{x,1})/d$;
 - 6: $y''_0 = (a_1 b_{y,2} - a_2 b_{y,1})/d$;
 - 7: $z''_0 = (a_1 b_{z,2} - a_2 b_{z,1})/d$;
 - 8: $\kappa = \|\mathbf{r}' \times \mathbf{r}''\|/\|\mathbf{r}'\|^3$;
 - 9: $\mathbf{T} = \mathbf{r}'/\|\mathbf{r}'\|$;
 - 10: $\mathbf{N} = \mathbf{r}'' - (\mathbf{r}'' \cdot \mathbf{T})\mathbf{T}$;
 - 11: $\mathbf{N} = \mathbf{N}/\|\mathbf{N}\|$;
 - 12: $\mathbf{B} = \mathbf{T} \times \mathbf{N}$;
-

(d) Torsion estimator for space curves

Similarly to the curvature estimation of space curves, the algorithm 1 has to be modified in order to also include the computation of a_4 , a_5 , a_6 , $b_{x,3}$, $b_{y,3}$ and $b_{z,3}$. Algorithm 5 then implements the torsion estimator. This algorithm provides an estimation for the curvature, for the torsion, and for the tangent, normal and binormal vectors.

(e) Boundary conditions

The above algorithms compute the curvature at a point \mathbf{p}_0 using $2q + 1$ samples around that point. However, for points close to the boundary of a curve, or in the specific conditions such as those of section 7 *Applications to compression*,

Algorithm 5 Independent coordinates method for space curves

- 1: call Set 3D Weighted Least Squares Variables;
- 2: call SOLVE

$$\begin{bmatrix} a_1 & a_2 & a_4 \\ a_2 & a_3 & a_5 \\ a_4 & a_5 & a_6 \end{bmatrix} \cdot \begin{bmatrix} x'_0 & y'_0 & z'_0 \\ x''_0 & y''_0 & z''_0 \\ x'''_0 & y'''_0 & z'''_0 \end{bmatrix} = \begin{bmatrix} b_{x,1} & b_{y,1} & b_{z,1} \\ b_{x,2} & b_{y,2} & b_{z,2} \\ b_{x,3} & b_{y,3} & b_{z,3} \end{bmatrix}$$

- 3: $\kappa = \|\mathbf{r}' \times \mathbf{r}''\|/\|\mathbf{r}'\|^3$;
 - 4: $\tau = -((\mathbf{r}' \times \mathbf{r}'') \cdot \mathbf{r}''')/\|\mathbf{r}' \times \mathbf{r}''\|^2$.
 - 5: $\mathbf{T} = \mathbf{r}'/\|\mathbf{r}'\|$;
 - 6: $\mathbf{N} = \mathbf{r}'' - (\mathbf{r}'' \cdot \mathbf{T})\mathbf{T}$;
 - 7: $\mathbf{N} = \mathbf{N}/\|\mathbf{N}\|$;
 - 8: $\mathbf{B} = \mathbf{T} \times \mathbf{N}$;
-

tion, those points cannot be centered on p_0 . In that case one can either reduce the width q of the sliding window, or simply compute the curvature using a non-centered window.

6 Experimental results

In this section the performance of the proposed algorithms will be compared to the previous works mentioned in section 3 *Previous and related works*.

(a) Experimental setting

The tests included here were performed on the three planar curves of Figure 14,15 and Figure 16, and on the five spatial curves of Figure 1. All of them were uniformly sampled in time. As a consequence, the samples were equally spaced for the circle, but not for the ellipse and the spiral. In the noisy case, as proposition 2 showed, it is important to keep σ/δ_1^2 small for the curvature estimation. Similarly, for the torsion estimation, the ratio σ/δ_1^3 must be small. Therefore, the noise was simulated as a uniform random variable in the normal segment (disc, in the 3D case) of radius $\sigma\bar{\mathbf{I}}^d$, where σ is fixed, d is the derivation order required and $\bar{\mathbf{I}}$ is the average distance between consecutive samples. The

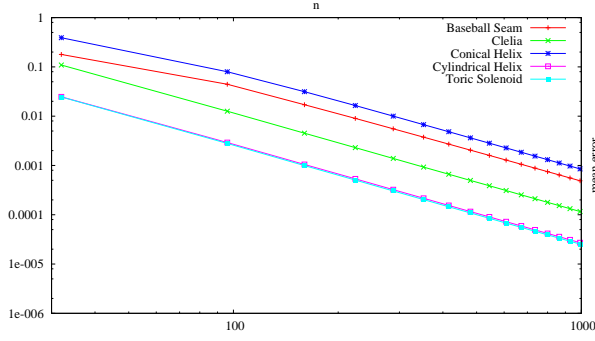
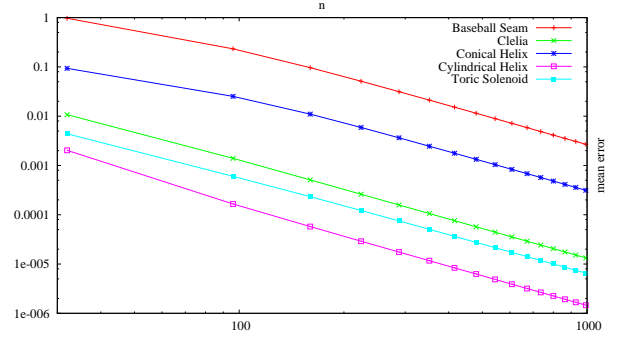
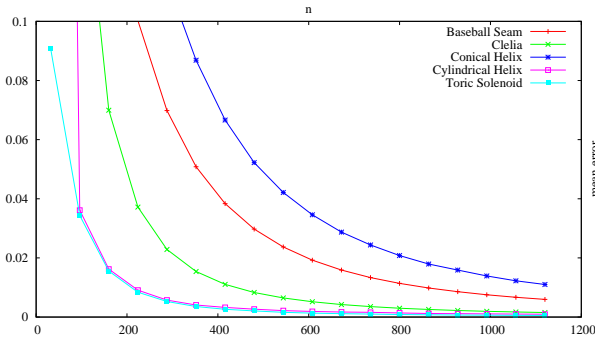
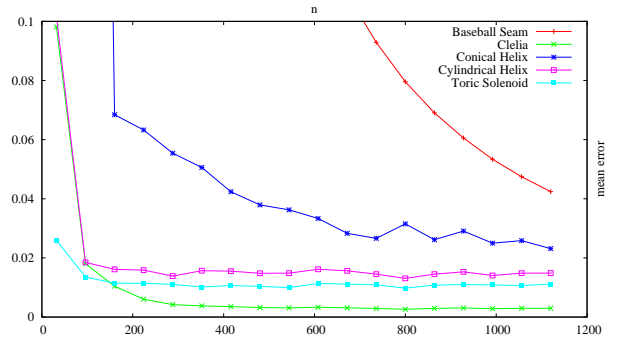
9(a): Curvature estimation without noise (logarithmic scale): $q = 2$.9(b): Torsion estimation without noise (logarithmic scale): $q = 2$.

Figure 9: Tests on spatial curves of Figure 1: convergence in the noiseless case (logarithmic scale) for the curvature and the torsion.

10(a): Curvature estimation with noise: $q = 9, \sigma = 1$.10(b): Torsion estimation with noise: $q = 9, \sigma = 1$.Figure 10: Tests on spatial curves of Figure 1: behaviour in the noisy case: with σ/δ_1^3 fixed, the curvature converges, and the torsion error stabilizes.

number of points in the window $2q + 1$ was fixed and the sample density was let to grow until the error become small.

The algorithm has been tested with weights given by the formula presented in section 4 *Theoretical framework* and also for constant weights.

Since the parametric formula was known for the examples tested, the real curvature was computed using automatic differentiation [10]. The absolute error between estimated curvature \hat{k} and the real value k at a point \mathbf{p}_j is measured by $E(\mathbf{p}_j) = \|\hat{\kappa}(p_j) - |\kappa(p_j)|\|$. The mean error was obtained as the arithmetic mean of the absolute errors at the considered points.

(b) Results

In the noiseless case ($\sigma = 0$), it was observed that, for all curves and methods considered, the punctual error $E(\mathbf{p}_j)$ increase with q (see Figure 14(c), Figure 15(c), and Figure 16(c)). This is not surprising, since the curvature estimation should be better when using points closer to the base point.

In the noisy case, it was observed that the use of more sample points can improve the estimates. The ideal number

of points q depends on the curve, on the point, on the sampling and on σ (see Figure 14(e), Figure 15(e) and Figure 16(e), Figure 14(g), Figure 15(g) and Figure 16(g)).

Consider now the error as a function of the number of samples with q and σ fixed. This is exactly the context of the convergence analysis. And the experimental results confirms the convergence analysis: the mean error is reduced when the distance between consecutive samples is reduced (see Figure 14(d), Figure 15(d) and Figure 16(d), Figure 14(f), Figure 15(f) and Figure 16(f), Figure 14(h), Figure 15(h) and Figure 16(h)).

The graphs of Figure 14, Figure 15 and Figure 16, and many other tests that have been performed show that the performance of the methods described above are close to the best among the methods tested, in a great variety of sampling and noise conditions. Also, the independent coordinates method has shown consistently better results than the dependent coordinates method.

In the 3D noiseless case, the mean error in the curvature estimation decreases to 0 (see Figure 9(a)), as the theory predicted. The same occurred for torsion, although theorem

does not include this case (see Figure 9(b)). Figure 10 shows the convergence of curvature and torsion estimators in the noisy case. Since the simulated noise has standard deviation $\sigma\bar{1}^3$, the theory predicts that the errors in the first and second derivatives tends to 0 when the number of samples points increases. This explains why the error in curvature estimation tends to 0 (Figure 10(a)) while the error in torsion estimation remains small but stable (Figure 10(b)). In the 3D case, the curvature and torsion estimators performed well in the sense that the obtained errors were small. Instead of comparing with other methods to confirm this assertion, the next section will introduce a practical framework to test those methods.

7 Applications to compression

Among the applications of curvature estimation methods, predictors for lossy geometric compression scheme provide real challenges. In those applications, the regularity of the geometric object to be compressed allows predicting the geometry of a vertex from the previously visited ones. Those schemes provide a very nice framework for testing the methods described above in a non-simulated noise environment, as the performance of the predictor can be directly measured from the final compression rate. The emphasis will be now put on two specific geometric predictors: the first one in 2D for Simplicial Isocontour Compression [13], and the second one in 3D for the Edgebreaker [21, 14].

(a) Prediction for Simplicial Isocontour Compression

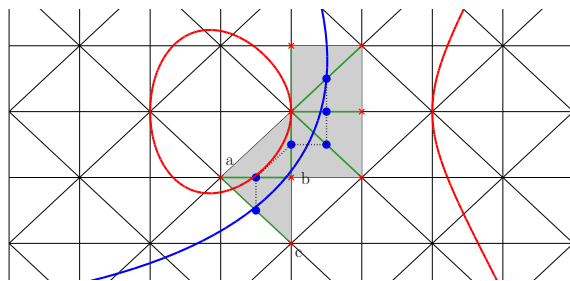


Figure 11: “Singular” curve: the predictor must guess whether the curve will traverse edge ac or bc , approximating the real curve (in red) by the scheme proposed above (blue curve) using only the points known to the decoder (in blue), i.e. midpoints of the edges crossing the curve.

This method aims at encoding a curve by its intersections with an adapted simplicial grid. The intersection are first transmitted in a hit/fail manner. Therefore, the decoder reconstructs the curve considering that it crosses always at the midpoint of the edges of the simplicial grid. When the decoded curve enters a triangle abc of the simplicial grid by the edge ab , the predictor must guess whether a curve crosses the edge ac or bc . The scheme proposed here computes the parabola approximating the curve around the

edge ab as described in section 5(b) *Independent coordinates method for planar curves*. The predictor considers that the curve crosses the same edge as the approximating parabola (see Figure 11).

The method of section 5(b) *Independent coordinates method for planar curves* is particularly well suited in that case: The predictor uses an extrapolation of the curve, which is not directly provided by 3–points methods. Moreover, whereas the circle fitting method provides a highly accurate approximation of the curvature, the osculating circle generically crosses the curve at the approximation point, which could lead to a bad prediction even in ideal conditions, as the curve must stay as close as possible to the curve. The independent coordinate method above gives very good results, with a mean above 80% of success on the benchmark of table 1.

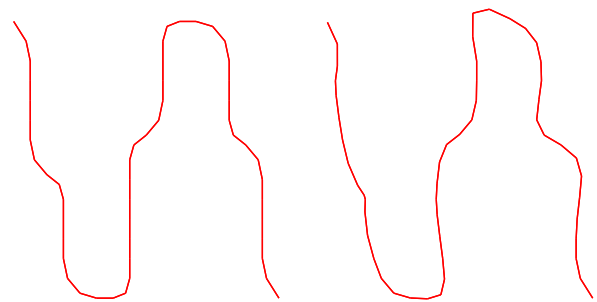


Figure 12: A decompressed implicit sine wave without smoothing (left) and with the predictor on–the–fly smoothing (right).

In a second step, the reconstruction of the curve can be enhanced by a smoothing operation, moving the intersection of the curve with the simplicial grid away from the edge midpoint. This should reduce the distortion of the decoded curve. In order to perform this operation on–the–fly, the approximation used by the predictor is directly used to compute the new intersection (see Figure 12). This smoothing behaves well when the curve is regular and well–sampled (see table 1).

(b) Prediction for the Edgebreaker

As most of the 3D surface encoders, the Edgebreaker algorithm compresses a triangulated surface by encoding its connectivity and the position of the vertices. To encode the coordinates of the vertices more efficiently, a predictor can be used to guess the position of a vertex from the former triangle in the compression traversal. The classical scheme relies on parallelogram prediction [23]. This scheme has been enhanced in many ways, but this section aims at testing the method of section 5(d) *Torsion estimator for space curves* more than defining another geometric predictor for the Edgebreaker. Therefore, the objective of the predictor

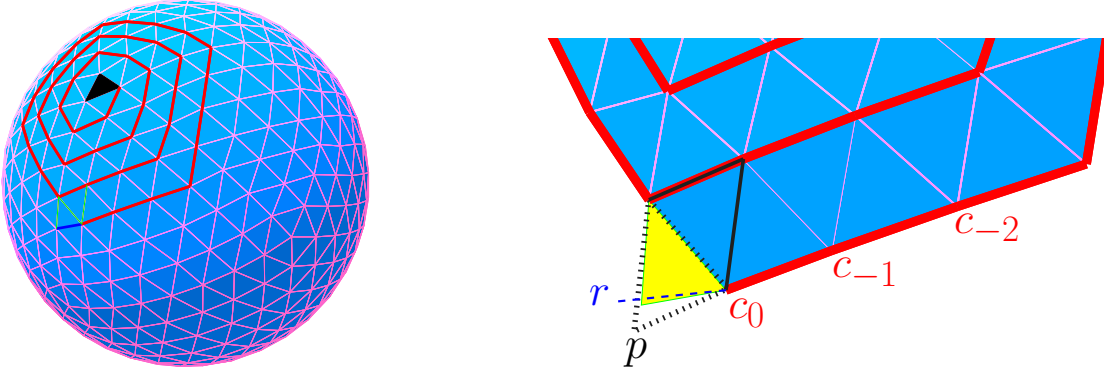


Figure 13: The Edgebreaker cuts the compressed surface along a curve in the space. An extrapolation of this curve is used to enhance the parallelogram predictor. The predictor uses the parallelogram predictor to guess the distance from the last vertex of the curve, and rotates this estimation according to the approximating curve.

Curve	Prediction	Size (bits)			Distortion (10^{-3})		
	success	direct	prediction	gain	direct	prediction	gain
bicorn	80.3 %	1479	1014	46 %	2.26	2.52	-10 %
circle	80.6 %	1658	1106	50 %	2.38	2.20	8 %
cubic	81.4 %	566	324	75 %	3.48	3.99	-13 %
ellipse	82.5 %	1430	920	55 %	2.03	1.99	2 %
singular	77.9 %	1501	1099	37 %	1.43	1.36	5 %
hyperbola	80.2 %	1648	1164	42 %	1.53	1.39	10 %
sinoide	84.5 %	2645	1612	64 %	0.91	0.76	20 %

Table 1: Results of the predictor for the Simplicial Isocontour Compression: for each one of the implicit curves, the number of successes over the erroneous guesses of the predictor is around 80%. The gain in the size of the compressed curve is half of the compressed size. The smoothing induced by the predictor reduces the distortion for most of the case. All the results were obtained as means of the same model over various resolution of the grid, from 50 to 3000 vertices, with $q = 84$.

# vertices	# models	//gram	curve	gain
0- 500	16	6.917	6.841	2.7 %
500- 1000	29	6.962	6.830	1.5 %
1000- 3000	39	4.519	4.454	2.2 %
3000-10000	42	2.118	2.077	2.1 %
10000-20000	15	0.867	0.870	-0.5 %
20000-50000	20	0.671	0.665	0.9 %

Table 2: Results of the predictor for the Edgebreaker: the results are compared in terms of mean error with the real vertex coordinates. The approximation is performed with $q = 12$. The distortion are in per-thousand.

is to enhance the parallelogram predictor by approximating the curve formed by the cut of the Edgebreaker (see Figure 13).

The predictor works as follows: The parallelogram-predicted point p is used to estimate the distance d of the unknown vertex. The approximation of the cut curve $\{c_{-q}, \dots, c_{-1}, c_0\}$ at parameter d gives another point r . The predicted point will then be the point e , image of p by the

rotation in the plane $\vec{n}, \overrightarrow{c_0 p}$ of arc d , where \vec{n} is the normal to the parallelogram (see Figure 13).

The approximation is considered coherent when r and c_{-1} are on the same side of the plane of the parallelogram. If the approximation is not coherent, the parallelogram predictor is used. This predictor enhances the parallelogram predictor (see table 2), validating the curve approximation of section 5(d) *Torsion estimator for space curves* in a real context.

8 Conclusion

This paper proposed curvature and torsion estimators for sampled curves in the plane and in the space. They are based on weighted least-squares fitting of parametric curves. It analysed the convergence of the curvature estimators under reasonable hypothesis.

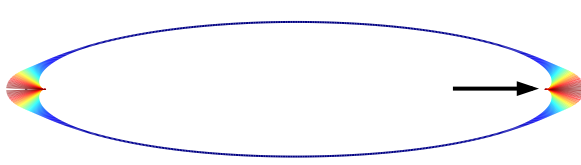
The proposed curvature estimators was compared experimentally with some important methods in the literature and their performance were close to the best. Based on experiments, it has also been observed that they are robust with respect to noise and that they behave well under a great variety of sampling conditions.

A very important advantage is that it can be immediately generalized for the estimation of curvature and torsion of spatial curves. Other advantage of the proposed method is that it can be easily implemented. The program used for comparison is available at [16].

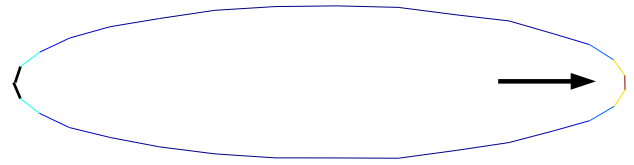
The authors plan to continue this work in order to develop predictors for other 3D compression schemes. Another natural extension, would be to develop a new estimator for geodesic curvature of curves on surface in the space.

References

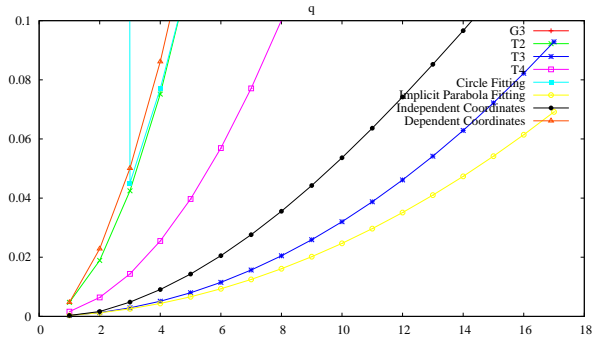
- [1] N. Amenta, S. Choi and R. Kolluri. The Power Crust, unions of balls, and the medial axis transform. *Computational Geometry: Theory and Applications*, 19(2–3):127–153, 2001.
- [2] A. Belyaev. Plane and space curves. curvature. curvature-based features. Max-Planck-Institut für Informatik, 2004.
- [3] H. H. Biscaro, A. Castelo and G. Nonato. A topological approach to curve reconstruction from scattered points. *Prépublicações do ICMC-USP*, 2004.
- [4] M. do Carmo. *Differential geometry of curves and surfaces*. Prentice Hall, 1976.
- [5] F. Cazals, F. Chazal and T. Lewiner. Molecular shape analysis based upon Morse–Smale complex and the Connolly function. In *Symposium on Computational Geometry*, pages 351–360. ACM, 2003.
- [6] D. Coeurjolly, M. Serge and T. Laure. Discrete curvature based on osculating circles estimation. In *Lecture Notes in Computer Science*, volume 2059, pages 303–312. Springer, 2001.
- [7] D. Coeurjolly and S. Svensson. Estimation of curvature along curves with application to fibres in 3D images of paper. In *Lecture Notes in Computer Science*, volume 2749, pages 247–254. Springer, 2003.
- [8] L. da Fontoura Costa and R. M. Cesar Jr. *Shape analysis and classification*. CRC Press, 2000.
- [9] L. F. Estrozi, L. G. R. Filho, A. G. C. Bianchi, R. M. Cesar Jr and L. da Fontoura Costa. 1D and 2D fourier-based approaches to numeric curvature estimation and their comparative performance assessment. *Digital Signal Processing*, 13:172–197, 2003.
- [10] A. Griewank. *Evaluating derivatives, principles and techniques of algorithmic differentiation*, volume 19 of *Frontiers in Applied Mathematics*. SIAM, 2000.
- [11] S. Gumhold. Designing optimal curves in 2d. In *CEIG*, pages 61–76, 2004.
- [12] P. Lancaster and K. Salkauskas. *Curve and surface fitting: an introduction*. Academic Press, 2002.
- [13] T. Lewiner, J. Gomes Jr, H. Lopes and M. Craizer. Arc-length based curvature estimator. In *Sibgrapi*, pages 250–257, Curitiba, Oct. 2004. IEEE.
- [14] T. Lewiner, L. Velho, H. Lopes and V. Mello. Hierarchical isocontours extraction and compression. In *Sibgrapi*, pages 234–241, Curitiba, Oct. 2004. IEEE.
- [15] T. Lewiner, J. Gomes Jr, H. Lopes and M. Craizer. Curvature estimation: theory and practice. Technical report, Department of Mathematics, PUC–Rio, 2004.
- [16] T. Lewiner, J. Gomes Jr, H. Lopes and M. Craizer. <http://www.mat.puc-rio.br/~fomlew/-curvature.zip>, 2004.
- [17] H. Lopes, J. Rossignac, A. Safonova, A. Szymczak and G. Tavares. Edgebreaker: a simple compression for surfaces with handles. In C. Hoffman and W. Bronsvort, editors, *Solid Modeling and Applications*, pages 289–296, Saarbrücken, Germany, 2002. ACM.
- [18] F. Mokhtarian and A. K. Mackworth. A theory for multiscale, curvature based shape representation for planar curves. *Transactions on Pattern Analysis and Machine Intelligence*, 14:789–805, 1992.
- [19] N. M. Patrikalakis and T. Maekawa. *Shape interrogation for computer aided design and manufacturing*. Springer, New York, 2002.
- [20] V. Pratt. Direct least-squares fitting of algebraic surfaces. *Computers & Graphics*, 21(4):145–152, 1987.
- [21] J. Rossignac. Edgebreaker: connectivity compression for triangle meshes. *Transactions on Visualization and Computer Graphics*, 5(1):47–61, 1999.
- [22] J. A. Sethian. *Fast marching methods and level set methods*. Cambridge Monograph on Applied and Computational Mathematics. Cambridge University Press, 1999.
- [23] C. Touma and C. Gotsman. Triangle mesh compression. In *Graphics Interface*, pages 26–34, 1998.
- [24] S. Utcke. Error-bounds on curvature estimation. *Lecture Notes in Computer Science*, 2695:657–666, 2003.
- [25] M. Worring and A. W. M. Smeulders. Digital curvature estimation. *CVGIP: Image Understanding*, 58(3):366–382, 1993.



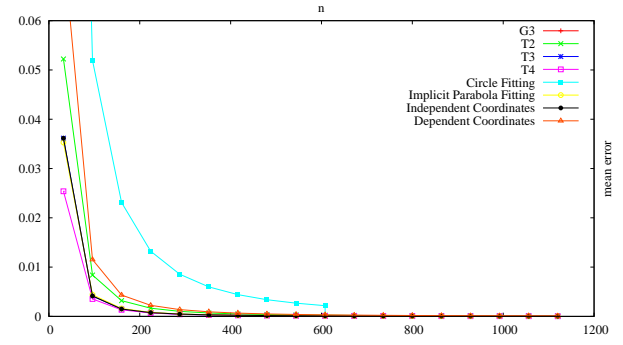
14(a): Ellipse without noise: $n = 1000, \sigma = 0$.



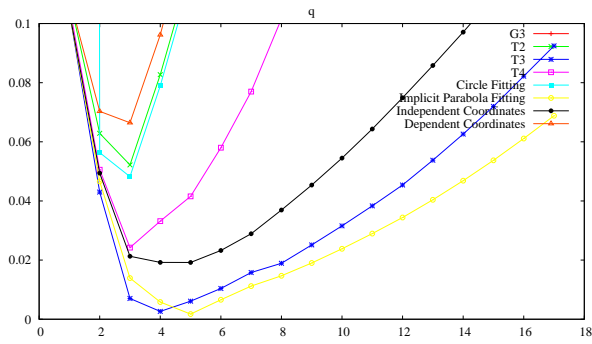
14(b): Ellipse with medium noise: $n = 32, \sigma = 0.5$.



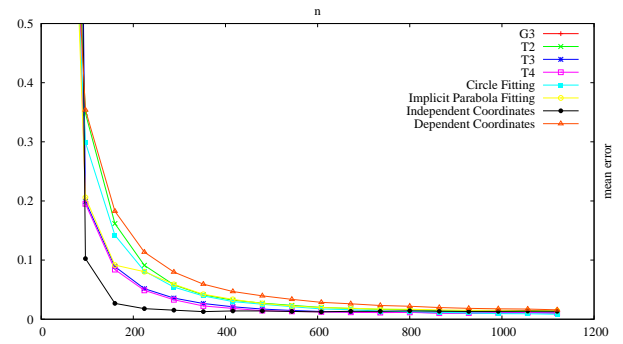
14(c): Punctual error, $n = 500, \sigma = 0$.



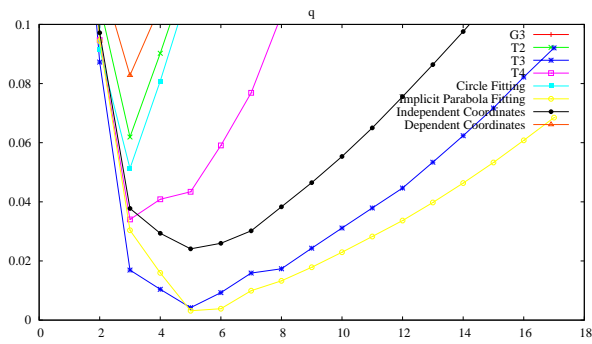
14(d): Mean error, $q = 1, \sigma = 0$.



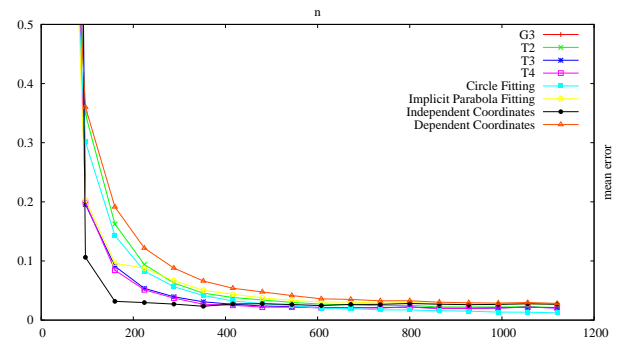
14(e): Punctual error, $n = 500, \sigma = 0.5$.



14(f): Mean error, $q = 10, \sigma = 0.5$.

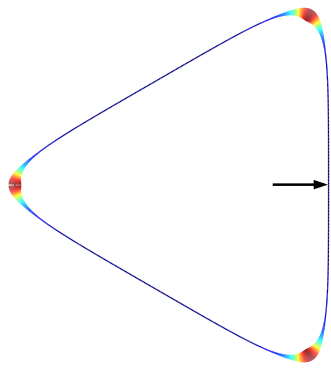


14(g): Punctual error, $n = 500, \sigma = 1$.

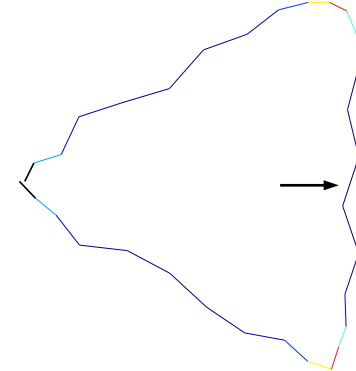


14(h): Mean error, $q = 10, \sigma = 1$.

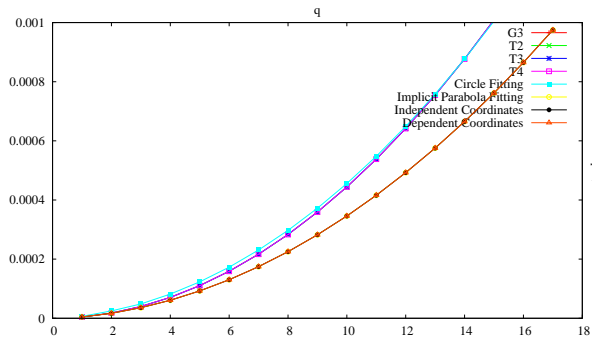
Figure 14: Tests on the ellipse $r(t) = (2 \cos(t), 0.5 \sin(t)) : t \in [-\pi, \pi]$: (a,b) the original curve noiseless and with medium noise. (c,e,g) punctual error with n fixed, varying q at the point pointed out on figures (a,b). (d,f,h) mean error with q fixed and varying n .



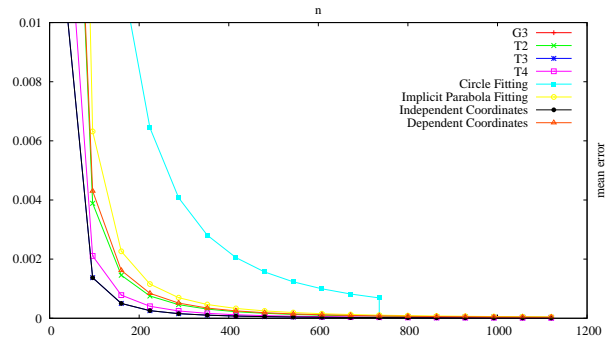
15(a): Hypocycloid without noise:
 $n = 1000, \sigma = 0.$



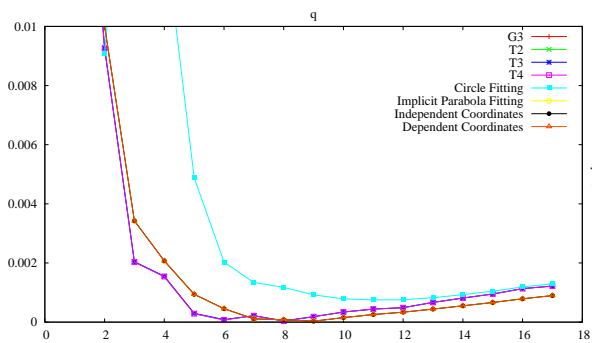
15(b): Hypocycloid with medium
noise: $n = 32, \sigma = 0.5.$



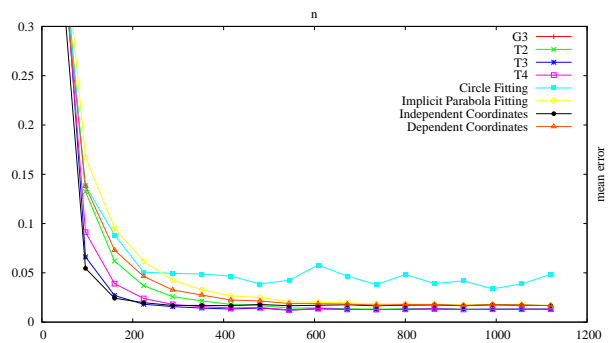
15(c): Punctual error, $n = 500, \sigma = 0.$



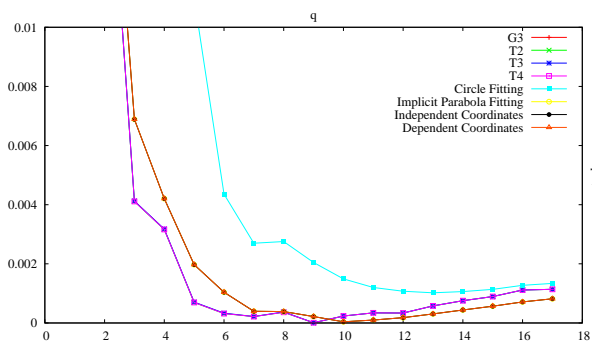
15(d): Mean error, $q = 1, \sigma = 0.5.$



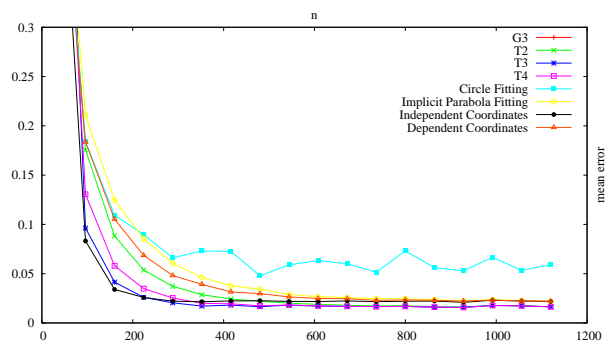
15(e): Punctual error, $n = 500, \sigma = 0.5.$



15(f): Mean error, $q = 10, \sigma = 0.5.$

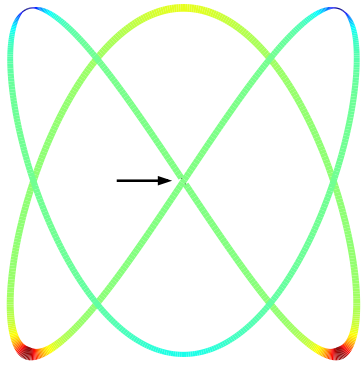


15(g): Punctual error, $n = 500, \sigma = 1.$

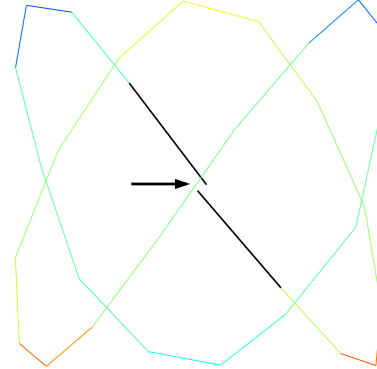


15(h): Mean error, $q = 10, \sigma = 1.$

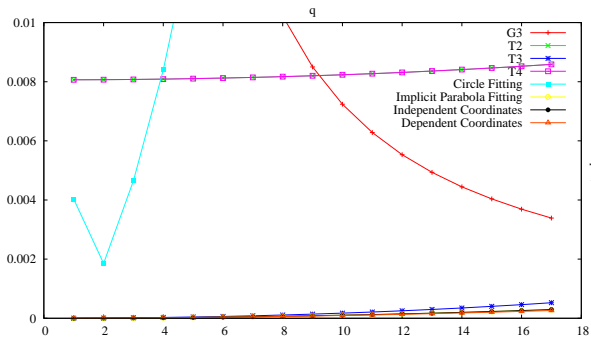
Figure 15: Tests on the hypocycloid $r(t) = (4 \cos(t) - \cos(2t), 4 \sin(t) + \sin(2t)) : t \in [-\pi, \pi]$: (a,b) the original curve noiseless and with medium noise. (c,e,g) punctual error with n fixed, varying q at the point pointed out on figures (a,b). (d,f,h) mean error with q fixed and varying n .



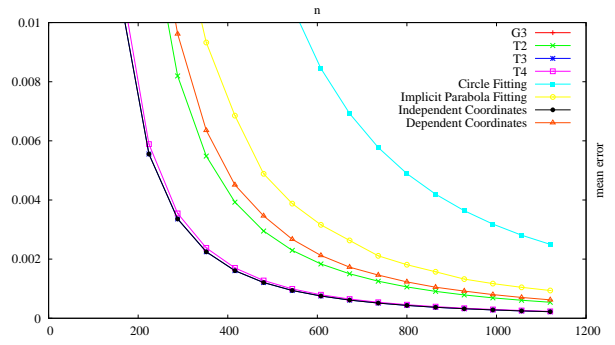
16(a): Lissajous without noise: $n = 1000, \sigma = 0$.



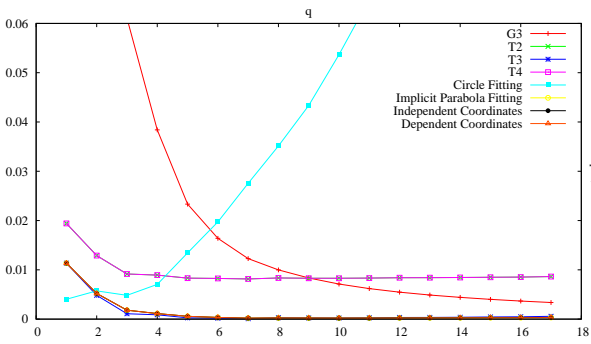
16(b): Lissajous with medium noise: $n = 32, \sigma = 0.5$.



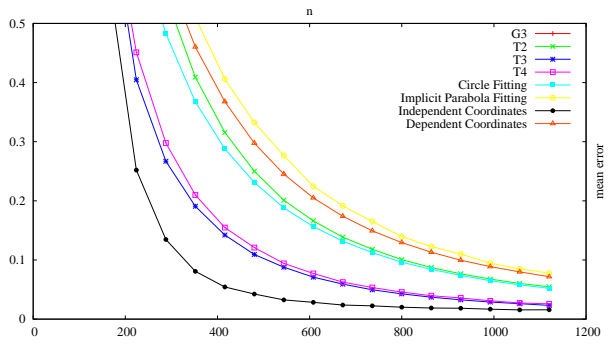
16(c): Punctual error, $n = 500, \sigma = 0$.



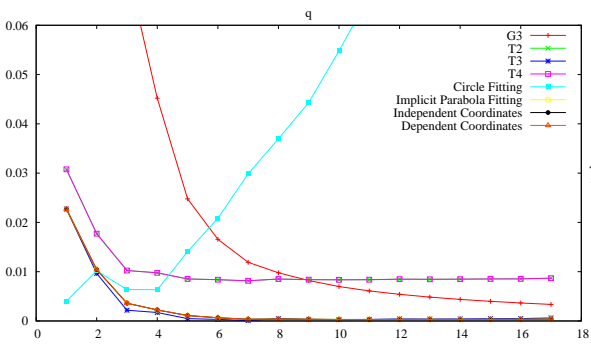
16(d): Mean error, $q = 1, \sigma = 0.5$.



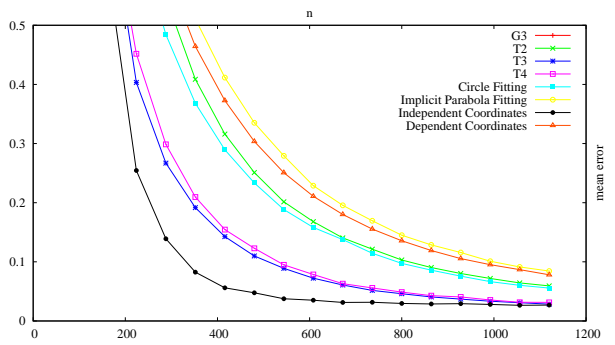
16(e): Punctual error, $n = 500, \sigma = 0.5$.



16(f): Mean error, $q = 10, \sigma = 0.5$.



16(g): Punctual error, $n = 500, \sigma = 1$.



16(h): Mean error, $q = 10, \sigma = 1$.

Figure 16: Tests on the lissajous $r(t) = (\sin(2t), \sin(3t)) : t \in [-\pi, \pi]$: (a,b) the original curve noiseless and with medium noise. (c,e,g) punctual error with n fixed, varying q at the point pointed out on figures (a,b). (d,f,h) mean error with q fixed and varying n .