Calculating the Cubic Bézier Arc Length by Elliptic Integrals

Hartmut Henkel, Oftersheim, Germany

14 May 2014

Contents

1 Motivation

Cubic Bézier curves are used by many graphics software tools as basis for the construction of more complex curves [Bartels et al. (1987)]. Each Bézier curve is defined by four control points: One at each end, plus two points controlling the direction of the curve from and towards either end point.

One parameter that is often of interest when designing a Bézier curve (or a more complex curve assembled by chaining of several such curves) is the resulting total arc length.

For arc length calculation of a curve or "*path*" the graphics design program MetaPost [Hobby, Hoekwater (2014)] uses a "*general bisection algorithm to subdivide the path until obtaining "well behaved" subpaths whose arc lengths can be approximated by simple means.*" Apparently this numerical, iterative method has some limitations in precision, which is a bit of a pity, since the general numeric precision of MetaPost has been significantly increased in the last few years.

Now two things have driven the experiment reported here: Firstly, the MetaPost source code looks quite opaque in this area, so delving into it seemed to be too tedious.

Secondly, there are statements in the net telling that Bézier arc length calculation, if tried analytically, will require the numerical solving of *elliptic integrals*. That made the second point somewhat interesting, since these integrals appear also in the context of digital IIR filter design [Orchard, Wilson (1997)], which is an interesting application in communication technology. For filter design there are many publications dealing with how to calculate elliptic integrals. So maybe it would not be too complicated to use these for the Bézier application as well?

Don't expect any mathematical depth or rigidity here. Everything is copied together from the literature in the internet, to make an experiment basically working, and to see how an arc length calculation by elliptic integrals would practically behave.

2 Bézier Curves

A cubic Bézier curve $z(t)$ in the plane is a parametric function of a variable (parameter) t, involving Bernstein polynomials $B_i^3(t)$ and four control points z_i [Bartels et al. (1987), Farin (2002)]:

$$z(t) = \sum_{i=0}^{3} B_i^3(t) z_i, \quad t \in [0, 1] \tag{1}$$

That is, the entire Bézier curve is travelled from point z_0 to point z_3, when t raises from 0 to 1. The control points z_i are normally given to shape the curve:

$$z_0 = (x_0, y_0), \qquad z_1 = (x_1, y_1), \qquad z_2 = (x_2, y_2), \qquad z_3 = (x_3, y_3) \tag{2}$$

The Bernstein polynomials are

$$B_i^3(t) = \binom{3}{i} t^i (1-t)^{3-i}, \quad i = 0 \dots 3$$

with binomial coefficients:

$$\binom{n}{i} = \begin{cases} \frac{n!}{i!(n-i!)} & \text{if} \quad 0 \leq i \leq n \\ 0 & \text{else} \end{cases}$$

The four required Bernstein polynomials are:

$$\begin{aligned} B_0^3(t) &= (1-t)^3 \\ B_1^3(t) &= 3t(1-t)^2 \\ B_2^3(t) &= 3t^2(1-t) \\ B_3^3(t) &= t^3 \end{aligned}$$

Putting these into Eqn. 1 gives:

$$z(t) = (1-t)^3 z_0 + 3t(1-t)^2 z_1 + 3t^2(1-t) z_2 + t^3 z_3$$

By expanding and sorting t by its powers, the coefficients of the parametric curve

$$\begin{aligned} x(t) &= a_3 t^3 + a_2 t^2 + a_1 t + a_0 \\ y(t) &= b_3 t^3 + b_2 t^2 + b_1 t + b_0 \end{aligned} \tag{3}$$

are found:

$$\begin{array}{rclrcl} a_0 & = & x_0, & b_0 & = & y_0 \\ a_1 & = & 3(x_1 - x_0), & b_1 & = & 3(y_1 - y_0) \\ a_2 & = & 3(x_0 - 2x_1 + x_2), & b_2 & = & 3(y_0 - 2y_1 + y_2) \\ a_3 & = & -x_0 + 3x_1 - 3x_2 + x_3, & b_3 & = & -y_0 + 3y_1 - 3y_2 + y_3 \end{array} \tag{4}$$

2.1 Bézier Arc Length

The arc length $L(t)$ of a general parametric curve in the plane is defined by:

$$L(t) = \int_0^t \sqrt{(x'(t))^2 + (y'(t))^2}\, dt \tag{5}$$

The derivatives from Eqn. 3 are:

$$\begin{array}{rcl} x'(t) & = & 3a_3t^2 + 2a_2t + a_1 \\ y'(t) & = & 3b_3t^2 + 2b_2t + b_1 \end{array}$$

Squaring these equations gives:

$$\begin{array}{rcl} (x'(t))^2 & = & (3a_3t^2 + 2a_2t + a_1)(3a_3t^2 + 2a_2t + a_1) \\ & = & 9a_3^2t^4 + 6a_3a_2t^3 + 3a_3a_1t^2 + 6a_3a_2t^3 \\ & & +\ 4a_2^2t^2 + 2a_2a_1t + 3a_3a_1t^2 + 2a_2a_1t + a_1^2 \\ & = & 9a_3^2t^4 + 12a_3a_2t^3 + 6a_3a_1t^2 + 4a_2^2t^2 + 4a_2a_1t + a_1^2 \\ (y'(t))^2 & = & 9b_3^2t^4 + 12b_3b_2t^3 + 6b_3b_1t^2 + 4b_2^2t^2 + 4b_2b_1t + b_1^2 \end{array} \tag{6}$$

So the radicand in Eqn. 5 involves t up to the 4th power. A polynomial of fourth order is used to sort this:

$$L(t) = \int_0^t \sqrt{c_4t^4 + c_3t^3 + c_2t^2 + c_1t^1 + c_0}\, dt \tag{7}$$

Comparison with Eqn. 6 gives the coefficients c_i:

$$\begin{array}{rcl} c_4 & = & 9(a_3^2 + b_3^2) \\ c_3 & = & 12(a_3a_2 + b_3b_2) \\ c_2 & = & 6(a_3a_1 + b_3b_1) + 4(a_2^2 + b_2^2) \\ c_1 & = & 4(a_2a_1 + b_2b_1) \\ c_0 & = & a_1^2 + b_1^2 \end{array} \tag{8}$$

Now the task is to solve the integral Eqn. 7.

3 Elliptic Integrals

The Eqn. 7 shows an elliptic integral, which can be read from various remarks where the calculation of Bézier arc lengths are discussed in the net. That the arc length computation by solving elliptic integrals is practically doable, has been shown [FitzSimons (1998)]. Checking the section no. 17 about Elliptic Integrals in [Abramowitz, Stegun (1972)] did not reveal any integral that would obviously fit to Eqn. 7, particularly since the square root expressions appear in the denominator, but not in the numerator. A further internet search suggested that the publications about elliptic integrals by Roland Bulirsch and Billie C. Carlson might provide more insight.

Indeed the paper [Carlson (1988)] shows many integrals "$[p]$" that follow a general scheme:

$$[p] = [p_1, \dots, p_n] = \int_y^x \prod_{i=1}^{n} (a_i + b_i t)^{p_i/2} dt$$

And with parameters

$$p_1 = p_2 = p_3 = p_4 = 1$$

this gives the integral of the Third Kind with a polynomial of 4th degree ("quartic case") as radicand:

$$\begin{aligned} [1,1,1,1] &= \int_y^x (a_1 + b_1 t^2)^{1/2} (a_2 + b_2 t)^{1/2} (a_3 + b_3 t)^{1/2} (a_4 + b_4 t)^{1/2} dt \\ &= \int_y^x \sqrt{(a_1 + b_1 t)(a_2 + b_2 t)(a_3 + b_3 t)(a_4 + b_4 t)} dt \end{aligned} \tag{9}$$

This integral can be solved by Eqn. (2.28) from [Carlson (1988)], which involves the numerical evaluation of "R-functions" R_J, R_D, R_C, and R_F. By the way, Carlson was also the editor of the Chapter 19 about Elliptic Integrals in the handbook [Olver et al. (2010)] and [NIST DLMF (2014)] to follow up [Abramowitz, Stegun (1972)].

The Eqn. 9 comes already rather near to Eqn. 7. The task would be to find the four real roots of the known radicand within Eqn. 7. However there are normally no real roots in the radicand of Eqn. 5, since it is a sum of two squares. Instead any "normal" roots will be conjugate complex ones, like $(t - a - ib)(t - a + ib) = (f + gt + ht^2)$. Happily a further search revealed the paper [Carlson (1992)], which covers the case of two pairs of conjugate complex zeros, included in the general case:

$$[p_1, p_2, p_2, p_1, p_5] = \int_y^x \prod_{i=1}^{2} (f_i + g_i t + h_i t^2)^{p_i/2} (a_5 + b_5 t)^{p_5/2} dt$$

With parameters

$$p_1 = p_2 = 1, \quad p_5 = 0$$

this gives the case $[1, 1, 1, 1]$, which looks as follows:

$$\begin{aligned} [1,1,1,1] &= \int_y^x (f_1 + g_1 t + h_1 t^2)^{1/2} (f_2 + g_2 t + h_2 t^2)^{1/2} dt \\ &= \int_y^x \sqrt{(f_1 + g_1 t + h_1 t^2)(f_2 + g_2 t + h_2 t^2)} dt \end{aligned} \tag{10}$$

The next step now is to find the two pairs of conjugate complex zeros in Eqn. 7 and put them into Eqn. 10. The paper [Carlson (1992)] further tells how to solve this integral numerically.

4 Root Finding

To find the coefficients in the radicand of Eqn. 10 from the c_i in Eqn. 8, a hopefully robust root finding algorithm was searched for.

Initially numerical root finder programs were checked. The commercial program used in [FitzSimons (1998)] was out of scope. The *Jenkins-Traub method* is suggested by [Press et al. (1992)] as "practically a standard in black-box polynomial root-finders", which sounded interesting. The corresponding paper [Jenkins, Traub (1970)] could be recovered and also the accompanying algorithm no. 493 [Jenkins, Traub (Source, 1970)], written in Fortran. This root finding algorithm was converted via `f2c` into C-code and a Lua interface was written for input and output data.

Another more recent root finder is the program MPSolve [Bini, Fiorentino (2000)]. The program source and documentation [Bini, Fiorentino (2000a)] were downloaded from the MPSolve homepage, and the sources were compiled into an executable. This is interfaced to Lua by writing the coefficients 8 into a data file, doing a `sys.execute()` call, and parsing the root data file from MPSolve.

Experiments with these numerical solvers showed convergence problems with "pathological" Bézier curves (like cusps). Since polynomials of up to 4th degree can still be solved analytically, an analytic root solver was searched for. The Fortran program `quartic.f` [Kraska (1998)] was tried, and this did find the roots faster and more stably, effectively bridging the gap between Eqn. 7 and 10. The source was converted to Lua, and this root finder was used for all further testing.

5 Test Setup

Now all ingredients of the experiment needed to be put together on the PC, running debian-Linux. Lua was chosen as programming language (`http://www.lua.org`).

Fortran listings for numerical calculation of the R-functions R_C and R_J are given in [Carlson (1988)], and for R_F and R_D they are in [Carlson (1987)], respectively. The source files available for download [Carlson, Notis (1981)] were not used, since these were older versions and not yet revised. The scanned sources were extracted from the PDF files and resurrected into Fortran code by use of the free OCR tool `gocr`, followed by manual corrections of OCR errors and re-formatting to the original code appearance. The program `f2c` was used for syntax and numerical checking. The Fortran sources were then manually converted to Lua code.

The formulas for numerical calculation of Eqn. 10 were typed in from [Carlson (1992)]; a rather tedious and funny list of variables needs to be calculated in sequence:

$$
\begin{aligned}
\xi_1 &= \sqrt{f_1 + g_1 x + h_1 x^2} \\
\xi_2 &= \sqrt{f_2 + g_2 x + h_2 x^2} \\
\eta_1 &= \sqrt{f_1 + g_1 y + h_1 y^2} \\
\eta_2 &= \sqrt{f_2 + g_2 y + h_2 y^2} \\
\xi_1' &= (g_1 + 2h_1 x)/(2\xi_1) \\
\eta_1' &= (g_1 + 2h_1 y)/(2\eta_1) \\
B &= \xi_1'\xi_2 - \eta_1'\eta_2 \\
E &= \xi_1'\xi_1^2\xi_2 - \eta_1'\eta_1^2\eta_2 \\
\theta_1 &= \xi_1^2 + \eta_1^2 - h_1(x-y)^2 \\
\theta_2 &= \xi_2^2 + \eta_2^2 - h_2(x-y)^2
\end{aligned}
$$

$$
\begin{aligned}
\zeta_1 &= \sqrt{(\xi_1+\eta_1)^2 - h_1(x-y)^2} \\
\zeta_2 &= \sqrt{(\xi_2+\eta_2)^2 - h_2(x-y)^2} \\
U &= (\xi_1\eta_2+\eta_1\xi_2)/(x-y) \\
M &= \zeta_1\zeta_2/(x-y) \\
\delta_{11} &= \sqrt{4f_1h_1 - g_1^2} \\
\delta_{22} &= \sqrt{4f_2h_2 - g_2^2} \\
\delta_{12} &= \sqrt{2f_1h_2 + 2f_2h_1 - g_1g_2} \\
\Delta &= \sqrt{\delta_{12}^4 - \delta_{11}^2\delta_{22}^2} \\
\Delta_+ &= \delta_{12}^2 + \Delta \\
\Delta_- &= \delta_{12}^2 - \Delta \\
L_+^2 &= M^2 + \Delta_+ \\
L_-^2 &= M^2 + \Delta_- \\
G &= 2\Delta\Delta_+ R_D(M^2, L_-^2, L_+^2)/3 + \Delta/(2U) \\
&\quad +(\delta_{12}^2\theta_1 - \delta_{11}^2\theta_2)/(4\xi_1\eta_1 U) \\
R_F &= R_F(M^2, L_-^2, L_+^2) \\
\Sigma &= G - \Delta_+ R_F + B \\
A(1,1,1,1) &= \xi_1\xi_2 - \eta_1\eta_2 \\
S &= (M^2+\delta_{12}^2)/2 - U^2 \\
\Lambda_0 &= \delta_{11}^2 h_2/h_1 \\
\Omega_0^2 &= M^2 + \Lambda_0 \\
\psi_0 &= g_1h_2 - g_2h_1 \\
X_0 &= -(\xi_1'\xi_2 + \eta_1'\eta_2)/(x-y) \\
\mu_0 &= h_1/(\xi_1\eta_1) \\
T_0 &= \mu_0 S + 2h_1h_2 \\
V_0^2 &= \mu_0^2(S^2 + \Lambda_0 U^2) \\
a_0 &= S\Omega_0^2/U + 2\Lambda_0 U \\
b_0^2 &= (S^2/U^2 + \Lambda_0)\Omega_0^4 \\
H_0 &= \delta_{11}^2\psi_0[R_J(M^2, L_-^2, L_+^2, \Omega_0^2)/3 + R_C(a_0^2, b_0^2)/2]/h_1^2 \\
&\quad -X_0 R_C(T_0^2, V_0^2) \\
[1,1,1,1] &= (\delta_{22}^2/h_2^2 - \delta_{11}^2/h_1^2)[\psi_0 H_0 + (\Lambda_0 - \delta_{12}^2)R_F]/8 \\
&\quad -(3\psi_0^2 - 4h_1h_2\delta_{12}^2)(\Sigma + \delta_{12}^2 R_F)/(24h_1^2h_2^2) \\
&\quad +[\Delta^2 R_F - \psi_0 A(1,1,1,1)]/(12h_1h_2) + E/(3h_1) \qquad (11)
\end{aligned}
$$

At the end the Bézier arc length integral, Eqn. 11 is calculated and multiplied by the parameter c_4 from Eqn. 8, which got lost in the root finding process.

After the integral calculations for R_J, R_D, R_C, and R_F had been separately tested, and all equations leading to Eqn. 11 had been carefully double-checked, the arc length calculation by the elliptic integral (Eqn. 11) worked out of the box, to quite some surprise (but see the next section).

For comparison the Bézier arc length was also calculated numerically by various means: The algorithm [Gravesen (1997)] using Bézier bisectioning was implemented in Lua. The numerical integration algorithm CQUAD from the GNU Scientific Library [Galassi et al. (2013)] also got a Lua interface; it runs with a preset max. relative error of 10^{-15}. Further, three integration loops were written in Lua, all running over 10^6 points, evaluating Eqn. 3 by simple summation of hypotenuse pieces, and integrating Eqn. 7 and Eqn. 10 by the extended trapezoidal rule, the latter to see the influence of the root finder.

6 Testing

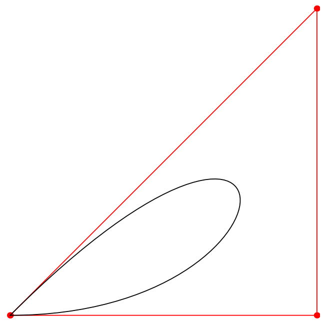
Testing was done from a Lua main file, which included all required library files. Only single Bézier arcs were used, not chains of several arcs. The goal was to see the effort to get the arc length right with an error of less than 10^{-10}.

The first example ist is a well-behaved Bézier arc with the following control points (in Lua notation):

$$\text{arc} = \{\{0,0\},\{10,0\},\{10,10\},\{0,0\}\}$$

The arc length results by the different methods have no numeric problems and agree well:

Eqn.	Method	Arc length	Steps
2	[Gravesen (1997)]	18.355664405651	55493
7	CQUAD integration	18.355664405651	591
3	Hypotenuse sum	18.355664405642	
7	Trapez. rule int.	18.355664405667	
10	Trapez. rule int.	18.355664405668	
11	[Carlson (1992)]	18.355664405651	
2	MetaPost	18.355680480222	

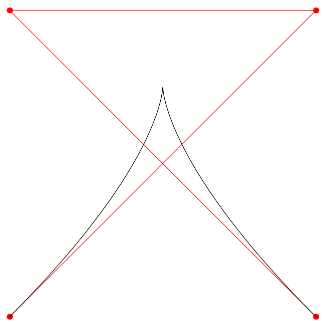


The second example shows a Bézier arc with a cusp:

$$\text{arc} = \{\{0,0\},\{10,10\},\{0,10\},\{10,0\}\}$$

Here the elliptic integral calculation fails.

Eqn.	Method	Arc length	Steps
2	[Gravesen (1997)]	18.284271247463	27102
7	CQUAD integration	18.284271247462	95047
3	Hypotenuse sum	18.28427124746	
7	Trapez. rule int.	18.284271247471	
10	Trapez. rule int.	18.284271247473	
11	[Carlson (1992)]	invalid arguments	
2	MetaPost	18.284272574347	

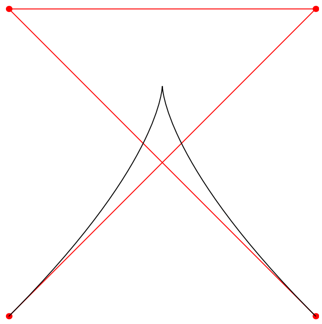


The third example is almost like the previous one, but it avoids the perfect cusp:

$$\text{arc} = \{\{0,0\},\{10,10\},\{0,9.999\},\{10,0\}\}$$

This is fine then with the elliptic integral calculation, but precision seems to be spoilt somehow in comparison to the [Gravesen (1997)] and CQUAD integration:

Eqn.	Method	Arc length	Steps
2	[Gravesen (1997)]	18.283649943205	27118
7	CQUAD integration	18.283649943204	102889
3	Hypotenuse sum	18.283649943187	
7	Trapez. rule int.	18.283649943231	
10	Trapez. rule int.	18.283650066623	
11	[Carlson (1992)]	18.283650122795	
2	MetaPost	18.283651786214	

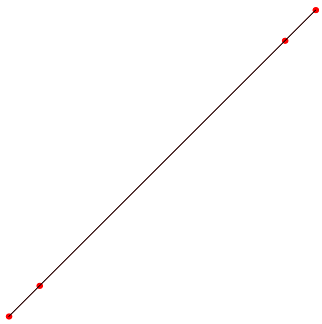


The fourth example shows a collinar case:

$$\text{arc} = \{\{0,0\},\{1,1\},\{9,9\},\{10,10\}\}$$

All methods are comfortable with this, the correct result is obviously $10 \cdot \sqrt{2}$:

Eqn.	Method	Arc length	Steps
2	[Gravesen (1997)]	14.142135623731	1
7	CQUAD integration	14.142135623731	33
3	Hypotenuse sum	14.142135623731	
7	Trapez. rule int.	14.142135623721	
10	Trapez. rule int.	14.142135623721	
11	[Carlson (1992)]	14.142135623731	
2	MetaPost	14.142143253125	

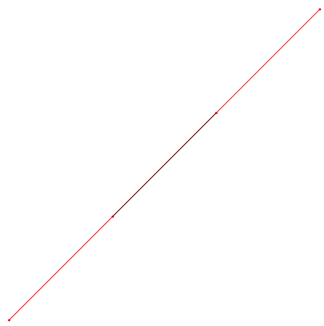


The fifth example has two cusps:

$$\text{arc} = \{\{0,0\},\{20,20\},\{-10,-10\},\{10,10\}\}$$

Here again the elliptic integral calculation fails:

Eqn.	Method	Arc length	Steps
2	[Gravesen (1997)]	26.791246264404	56
7	CQUAD integration	26.791246264404	187349
3	Hypotenuse sum	26.791246264389	
7	Trapez. rule int.	26.791246264474	
10	Trapez. rule int.	26.791246264473	
11	[Carlson (1992)]	R_D hangs	
2	MetaPost	26.791253893799	



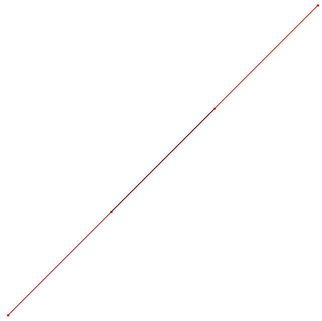
The sixth example is like the previous one, only with a slight shift of the 3rd control point:

$$\text{arc} = \{\{0,0\},\{20.001,20\},\{-10,-10\},\{10,10\}\}$$

In this case the input parameters for the integral R_J are outside the permissible range as given in the paper. The latter looked bad at first, since it would mean that elliptic integrals would only be suitable to calculate well-behaved Bézier arcs.

But then two papers [Gustafson (1982), Carlson (1993)] were found, which describe, how the R-functions can be replaced by approximation formulas for cases when the input parameters differ by orders of magnitude. E. g., by using the R_J approximation formula from [Carlson (1993)], an integral result is found also for this sixth example:

Eqn.	Method	Arc length	Steps
2	[Gravesen (1997)]	26.791625779745	933
7	CQUAD integration	26.791625779552	191231
3	Hypotenuse sum	26.79162577952	
7	Trapez. rule int.	26.791625779622	
10	Trapez. rule int.	26.791625779624	
11	[Carlson (1992)]	26.791625779944	
2	MetaPost	26.791633411815	



Six examples are enough for now.

6.1 Observations

The numerical method [Gravesen (1997)], which uses the Bézier subdivision from Paul de Casteljau, performes well for any curve shape (e. g., no problem with cusps). But it needs up to millions of Bézier subdivisions to reach the aspired precision.

The CQUAD integration works hard, needing many iterations when there are cusps, but it still reaches a sound result.

The simple numeric integrations provide reasonable results, but with 10^6 steps for one arc they are very slow.

The elliptic integral calculation works in general rather precisely for "well-behaved" Bézier curves, but it fails with hard cusp and collinear cases, and with cases near to these. Some of the R-functions then complain about underflows of their input variables. A few of these conditions can be remedied by use of the approximation formulas given in [Carlson (1993)] as proxies before the R-function calls.

7 Conclusion

This experiment would not have been possible without spotting the paper [Carlson (1992)], which so clearly gives a recipe how to numerically solve the required elliptic integral.

The experiment's result is, that elliptic integrals can be used in practice for precise Bézier arc length calculation; this has been shown [FitzSimons (1998)] already some time ago. Since the required R-functions converge quickly, the method seems to be rather fast.

A problem is, that the R-functions might receive input variables outside their permissible range. Here the approximation formulas [Carlson (1993)] come to an aid for some, but not for all cases. More and difficult work would be needed to ruggedize the arc length calculation method by elliptic integrals, to make it usable for general Bézier curves. This might involve the need to distinguish between a large number of cases, and it is not clear if this could be done with reasonable effort. It is also unclear how one could predict the practically achieved arc length precision for a given Bézier curve.

Finally, looking deeper into the arc length calculation method used by MetaPost might yield an easier improvement in arc length precision there.

References

[Abramowitz, Stegun (1972)] Handbook of Mathematical Functions. Edited by Milton Abramowitz and Irene A. Stegun. Dover Publications, Inc., New York, December 1972.

[Bartels et al. (1987)] An Introduction to Splines for Use in Computer Graphics & Geometric Modeling. Richard H. Bartels, John C. Beatty, Brian A. Barsky. Morgan Kaufman Publishers, Inc., Los Altos, California, 1987.

[Bini, Fiorentino (2000)] Design, Analysis, and Implementation of a Multiprecision Polynomial Rootfinder. D. A. Bini and G. Fiorentino. Numerical Algorithms, Volume 23, 2000, Pages 127–173.

[Bini, Fiorentino (2000a)] Numerical Computation of Polynomial Roots Using MPSolve Version 2.2. Dario Andrea Bini and Giuseppe Fiorentino. January 2000. From the MPSolve home page at http://www.dm.unipi.it/cluster-pages/mpsolve/.

[Carlson, Notis (1981)] Algorithm 577: Algorithms for Incomplete Elliptic Integrals. B. C. Carlson and Elaine M. Notis. ACM Transactions on Mathematical Software (TOMS), Volume 7, Issue 3, Sept. 1981. Algorithm 577, File 577.gz, at http://dl.acm.org/citation.cfm?id=355970 and http://www.netlib.org/toms/577.

[Carlson (1987)] A Table of Elliptic Integrals of the Second Kind. B. C. Carlson. Mathematics of Computation, Volume 49, Number 180, October 1987, Pages 595–606. File S0025-5718-1987-0906192-1.pdf at http://www.ams.org.

[Carlson (1988)] A Table of Elliptic Integrals of the Third Kind. B. C. Carlson. Mathematics of Computation, Volume 51, Number 183, July 1988, Pages 267–280. File S0025-5718-1988-0942154-7.pdf at http://www.ams.org.

[Carlson (1992)] A Table of Elliptic Integrals: Two Quadratic Factors. B. C. Carlson. Mathematics of Computation, Volume 59, Number 199, July 1992, Pages 165–180. File S0025-5718-1992-1134720-4.pdf at http://www.ams.org.

[Carlson (1993)] Asymptotic Approximations for Symmetric Elliptic Integrals. B. C. Carlson and John L. Gustafson. arXiv:math/9310223v1, 7 Oct 1993.

[Carlson, FitzSimons (2000)] Reduction theorems for elliptic integrands with the square root of two quadratic factors. B. C. Carlson and James FitzSimons. Journal of Computational and Applied Mathematics 118 (2000) 71–85.

[Carlson (2002)] Three Improvements in Reduction and Computation of Elliptic Integrals. B. C. Carlson. Journal of Research of the National Institute of Standards and Technology, Volume 107, Number 5, September–October 2002.

[Farin (2002)] Curves and Surfaces for CAGD — A Practical Guide. Gerald Farin. Morgan-Kaufman, 5th ed., 2002.

[FitzSimons (1998)] The length of a Bézier curve is an elliptic integral. Jim FitzSimons. 16 June 1998 (file date). http://www.tinaja.com/glib/bezlenjf.pdf.

[Galassi et al. (2013)] GNU Scientific Library, Reference Manual. Mark Galassi, Jim Davies, James Theiler, Brian Gough, Gerard Jungman, Patrick Alken, Michael Booth, Fabrice Rossi, Rhys Ulerich. Edition 1.16, for GSL Version 1.16, 17 July 2013.

[Gravesen (1997)] Adaptive subdivision and the length and energy of Bézier curves. Jens Gravesen. Computational Geometry 8 (1997) 13–31.

[Gray (2001)] Automatic Reduction of Elliptic Integrals Using Carlson's Relations. Norman Gray. Mathematics of Computation, Volume 71, Number 237, 22 July 2001, Pages 311–318. File S0025-5718-01-01333-3.pdf at `http://www.ams.org`.

[Gustafson (1982)] Asymptotic formulas for elliptic integrals. John L. Gustafson. Dissertation, Iowa State University, Ames, Iowa, 1982.

[Herbison-Evans (2011)] Solving Quartics and Cubics for Graphics. Don Herbison-Evans. Technical Report TR94-487. Basser Department of Computer Science, University of Sydney, Australia, updated 31 March 2011.

[Hobby, Hoekwater (2014)] MetaPost 1.990 (TeX Live 2014/dev) (kpathsea version 6.2.0dev). Original author of MetaPost: John Hobby. Author of the CWEB MetaPost: Taco Hoekwater.

[Jenkins, Traub (1970)] A Three-Stage Algorithm for Real Polynomials Using Quadratic Iteration. M. A. Jenkins and J. F. Traub. SIAM J. Numer Anal., Vol. 7, No. 4, December 1970.

[Jenkins, Traub (Source, 1970)] Subroutine RPOLY (Fortran). M. A. Jenkins and J. F. Traub. Algorithm 493, File 493.gz, at `http://dl.acm.org/citation.cfm?id=355643` and `http://www.netlib.org/toms/493`.

[Kraska (1998)] Subroutines for solving cubic, quartic and quintic equations. T. Kraska. Program `quartic.f`, download from `http://van-der-waals.pc.uni-koeln.de/quartic/quartic.html`.

[NIST DLMF (2014)] NIST Digital Library of Mathematical Functions. `http://dlmf.nist.gov/`.

[Olver et al. (2010)] NIST Handbook of Mathematical Functions. Edited by Frank W. J. Olver, Daniel W. Lozier, Ronald F. Boisvert, and Charles W. Clark. NIST National Institute of Standards and Technology, 2010.

[Orchard, Wilson (1997)] Elliptic Functions for Filter Design. H. J. Orchard, Alan N. Willson, Jr. IEEE Transactions on Circuits and Systems—I: Fundamental Theory and Applications, Vol. 44, No. 4, April 1997, pp. 273–287.

[Press et al. (1992)] Numerical Recipes in C — The Art of Scientific Computing. William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. Cambridge University Press, Second Edition, 1992.