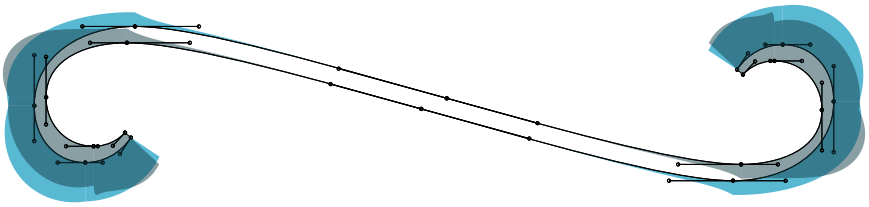
Curvatura — A Plug-In For FontForge

Linus Romer, linus.romer@gmx.ch

June 12, 2020



1 Introduction

Curvatura is a FontForge plug-in to harmonize, tunnify or add inflection points to the selected parts.

2 Prerequisites

On Linux you must have installed Python along with FontForge. On Windows FontForge embeds an own version of Python. Hence, you do not have to install Python additionally. FontForge has to be at least the version from May, 26 2019.
If you are on Ubuntu/Xubuntu/Kubuntu 18.04 you may consider a manual update:

```
sudo add-apt-repository ppa:fontforge/fontforge
sudo apt remove libgdraw5 libfontforge1 fontforge-common fontforge python-fontforge
sudo apt install libgdraw4=20190413-27-g1acfefa-0ubuntu1~bionic \
libfontforge1=20190413-27-g1acfefa-0ubuntu1~bionic \
fontforge-common=20190413-27-g1acfefa-0ubuntu1~bionic \
fontforge=20190413-27-g1acfefa-0ubuntu1~bionic \
python-fontforge=20190413-27-g1acfefa-0ubuntu1~bionic
```

If you are bound to an older version of FontForge, you may try harmonize-tunnify-inflection instead, which should work with older versions of FontForge too (but is not being developped anymore).

3 Installation

Copy the file Curvatura.py to `~/.config/fontforge/python` on Linux or to

`C:\Users\[YOUR USERNAME HERE]\AppData\Roaming\FontForge\python`

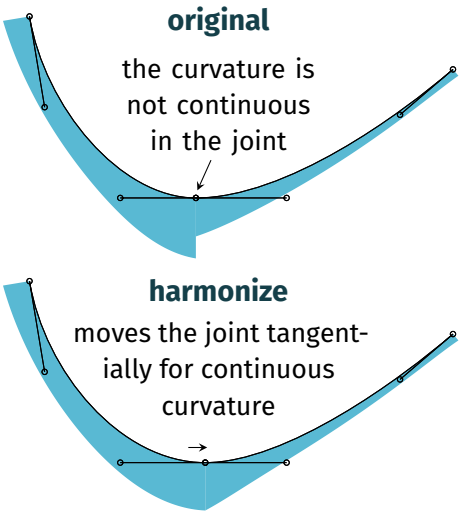
on Windows.

If you want to use hotkeys as well, you can replace the hotkeys file in the parent directory by the hotkeys file of this repository.

4 New Tools Added By Curvatura

After installation, FontForge will show in the Tools menu 4 new entries: «Harmonize», «Harmonize handles» , «Tunnify (balance)» and «Add points of inflection». Their effects will be described by examples. Details of the harmonizing algorithms can be found in section 6.

4.1 The *Harmonize* Tool For Cubic Béziers

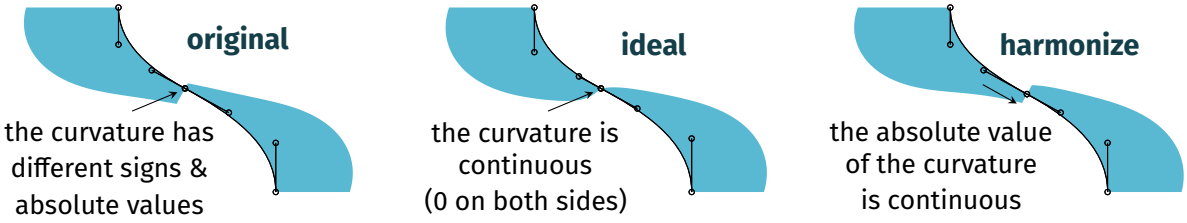


The «Harmonize» tool works on smooth nodes between two Bézier segments, as depicted to the right. The blue area shows the curvature comb. As one can see, the curvature is not continuous in the joint node (i.e. the curvature to the left is not the same as to the right).

«Harmonize» now moves the node between its handles to the position, where the curvature becomes continuous. In mathematics this is called G^2–continuous.

Fun fact: The joint node divides the line consisting of its handles in the ratio of the geometric mean of the distances of the other handles to the tangent. See subsection 6.1 for the proof.

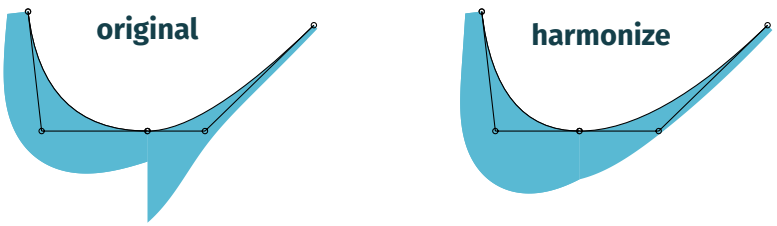
If the joint node is an inflection point (i.e. the curvature changes signs), the ideal mathematical solution would be to extend the non–attached handles to the joint tangent. Then the curvature would be zero on both sides and therefore continuous. In the generic case there is no possibility to move the joint node between its attached handles such that this condition is met.



Hence, «Harmonize» just moves the joining node between the handles until the curvature reaches the same *absolute value* on both sides. Then the curvature is *not continuous*, but at least the absolute value of the curvature is continuous. I.e. the curves change with the same «abruptness» on both sides.

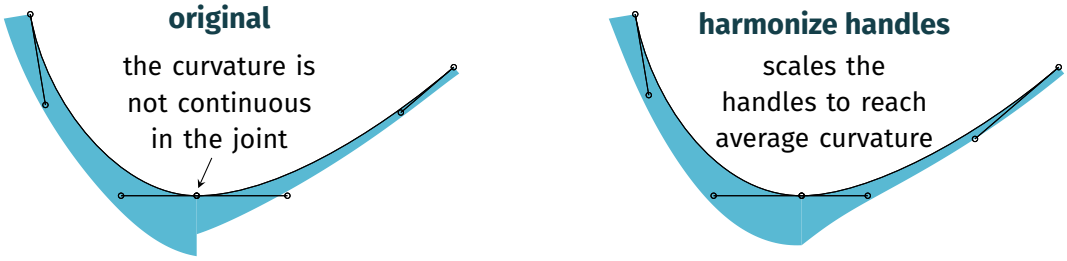
4.2 The *Harmonize* Tool For Quadratic Béziers

The «Harmonize» tool uses for quadratic Bézier splines a similar algorithm as for cubic Bézier splines (see subsection 6.2). As it is a iterated algorithm, results may change slightly if applied two times (but this happens only rarely).



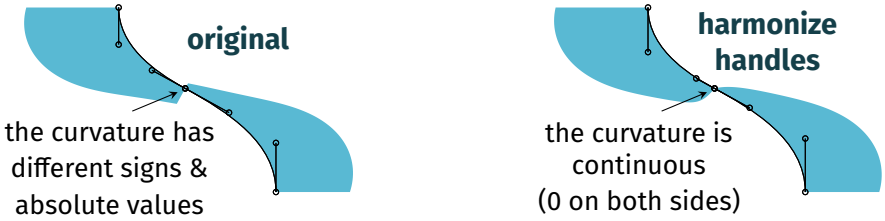
4.3 The *Harmonize handles* Tool

The «Harmonize handles» tool works on smooth nodes between two Bézier segments – just as «Harmonize». The difference is that not the node is moved but rather the handles are scaled such that the curvature in the joint becomes the average of the previous curvatures (see subsection 6.3 for mathematical details). The curvatures of the neighbour nodes are maintained.



This algorithm includes solving a quartic equation which may not always have a solution. Hence, this algorithm is iterated, which increases the chance of being successful. After applying «Harmonize handles» sucessfully, the curvature becomes continuous (G^2–continuous).

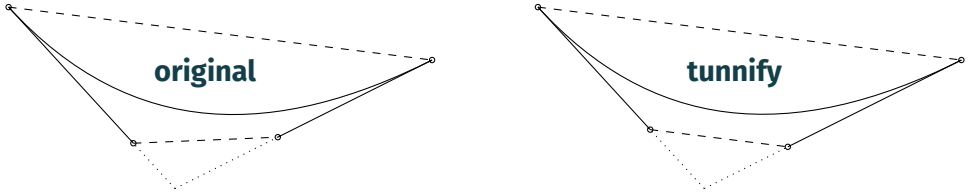
If the joint node is an inflection point (i.e. the curvature changes signs), the non–attached handles are extendended to the joint tangent. Then the curvature becomes zero on both sides of the joint and therefore continuous. *Caution:* After rounding the handles may exceed the tangent triangle, which means that an additional inflection point occurs on the segment!



Since the handles of quadratic Bézier splines cannot be scaled, this tool has no effect on them.

4.4 The *Tunnify* Tool

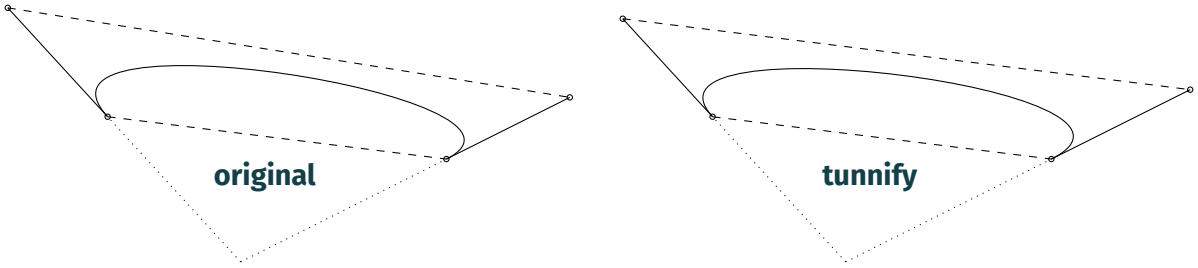
«Tunnify (balance)» moves the handles of a cubic Bézier segment such that the line between the handles is parallel to the line between the nodes. In the generic case (as depicted below) this is done by setting the control points to the arithmetic mean of their old ratios with respect to the tangent intersection point.



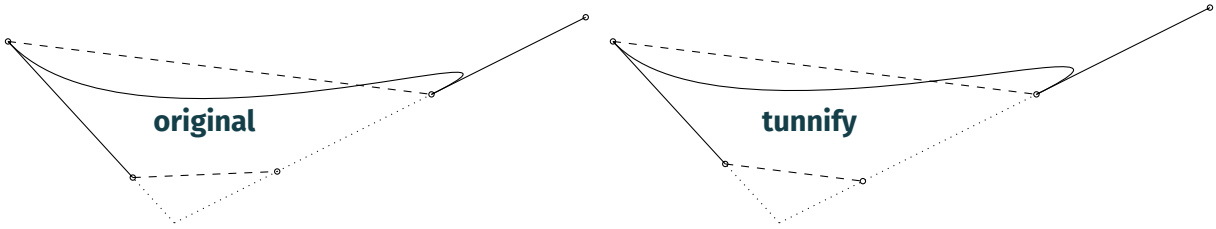
This tool...

- is advantageous if you are working with several masters, as it makes them more consistent.
- does not necessarily improve the curvatures (use one of the *harmonize* variants instead).
- does not have any meaning for quadratic Bézier splines.

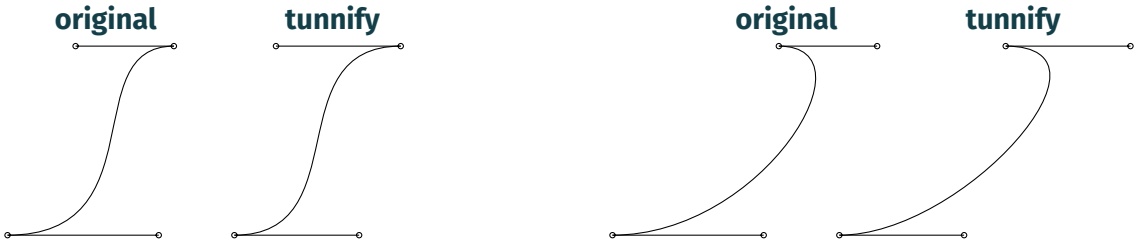
There are some special cases that are not handled in other programs. The first special case (the handles head away from the tangent intersection) can still be treated by the same rule as the generic case:



If the handles are not parallel and the control points lie on different sides of the straight line between the end nodes, one handle is mirrored with respect to the node it is attached to. Then the generic tunnify action is performed and finally the one handle is mirrored back:



If the handles are parallel, their lengths will be set to their arithmetic mean:



4.5 The *Add Points Of Inflection* Tool

«Add points of inflection» adds points of inflection (FontForge can natively display them but not natively add them). The curvature and the shape of the curves are not changed by this action:

As quadratic Bézier splines never have points of inflection, this tool has no effect on them.

5 Use Curvatura.py In Command Line

You can use `Curvatura.py` in the command line. It will harmonize all glyhps in a font and needs exactely 2 arguments:

```
python Curvatura.py input_font_file output_font_file
```

The input font file and output font file can be any file formats that FontForge understands (e.g. *.sfd, *.otf, *.ttf, *.pfb).

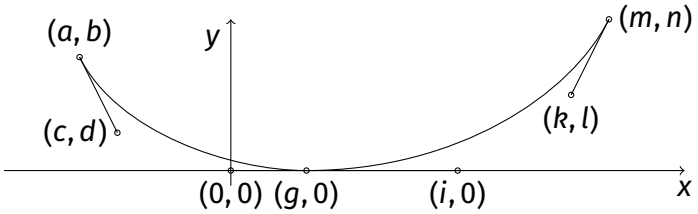
6 Algorithms To Make Two Adjacent G^1–Continuous Cubic Bézier Curves G^2–Continuous

6.1 G^2–continuity For Cubic Bézier Curves By Moving On-Curve Nodes Tangentially

Given two G^1–continuous cubic Bézier curves

$$\begin{pmatrix}x\\y\end{pmatrix} = (1-t)^3\begin{pmatrix}a\\b\end{pmatrix} + 3t(1-t)^2\begin{pmatrix}c\\d\end{pmatrix} + t^3\begin{pmatrix}g\\0\end{pmatrix} \quad \text{and} \quad \begin{pmatrix}x\\y\end{pmatrix} = (1-t)^3\begin{pmatrix}0\\0\end{pmatrix} + 3t(1-t)^2\begin{pmatrix}i\\0\end{pmatrix} + 3t^2(1-t)\begin{pmatrix}k\\l\end{pmatrix} + t^3\begin{pmatrix}m\\n\end{pmatrix}$$

with $0 < g < i$ we want to determine g such that the curves are G^2–continuous in their joint $(g, 0)$.



For the G^2–continuity at their joint $(g, 0)$ we have to equalize the curvature on both sides:

$$\frac{2d}{3g|g|} = \frac{2l}{3(i-g)|i-g|}$$

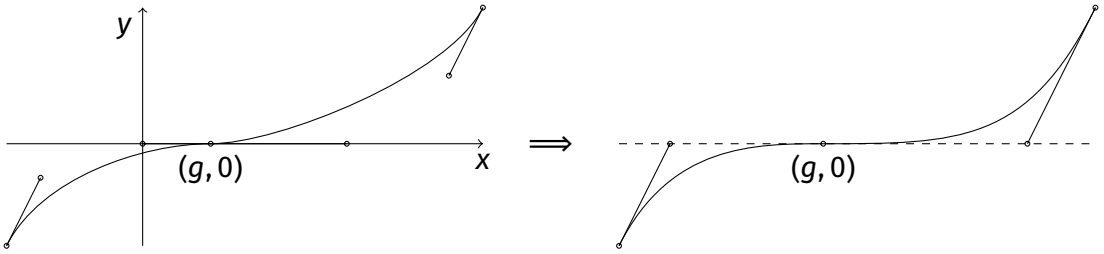
Because of $0 < g < i$, this simplifies to $\frac{d}{g^2} = \frac{l}{(i-g)^2}$. In the special case $d = l$, we have $g = i - g$. Solving for g we get

$$\boxed{g = \begin{cases}\frac{(d-\sqrt{dl})}{d-l} \cdot i & \text{if } d \neq l,\\ \frac{i}{2} & \text{else.}\end{cases}} \quad \text{or} \quad g = \begin{cases}\frac{(d+\sqrt{dl})}{d-l} \cdot i & \text{if } d \neq l,\\ \frac{i}{2} & \text{else.}\end{cases}$$

If we assume $d \geq 0$, and $l \geq 0$ (which is the generic case in typedesign) the second solution can lead to cases where $g < 0$, whereas the first solution always lead to $g \geq 0$ (remember that the geometric mean $\sqrt{dl}$ lies between d and l). Therefore, the algorithm in *Curvatura* uses the first solution. The peculiar thing is, that the ratio $\frac{(d+\sqrt{dl})}{d-l}$ is the ratio of the geometric mean $\sqrt{dl}$ between d and l!

6.1.1 Special Case: Joint Node Is Inflection Point

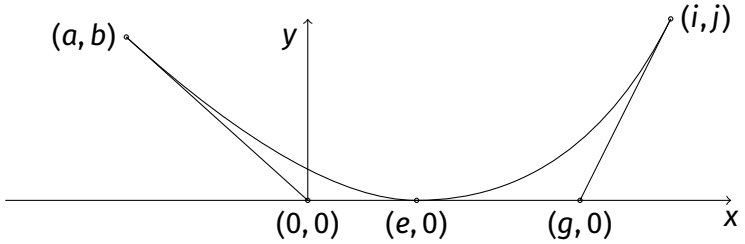
If the joint $(g, 0)$ is an inflection point (i.e. the curvature of both segments have different signs there), G^2–continuity implies that the curvature must be 0 at $(g, 0)$. This can only be satisfied if the control points lie on the tangent in $(g, 0)$:



However, such a treatment would be inconsistent compared to the generic case where only the joint node is moved. Therefore, *Curvatura* handles inflection point by mirroring one part with regards to the tangent in $(g, 0)$, applies the generic formula and finally mirrors the part back. In this way, the curvature in $(g, 0)$ is the same on both sides but with different signs.

6.2 G^2–Continuity For Quadratic Bézier Curves By Moving On-Curve Nodes Tangentially

Given two G^1–continuous quadratic Bézier curves $\binom{x}{y} = (1-t)^2\binom{a}{b} + t^2\binom{e}{0}$ and $\binom{x}{y} = (1-t)^2\binom{e}{0} + 2t(1-t)\binom{g}{0} + t^2\binom{i}{j}$ with $0 < e < g$ we want to determine e such that the curves are G^2–continuous in their joint $(e, 0)$.



For the G^2–continuity at their joint $(0, 0)$ we have to equalize the curvature on both sides:

$$\frac{b}{2e^2} = \frac{e}{2(g-e)^2}$$

which yields to

$$\boxed{e = \frac{(b - \sqrt{bj})}{b-j} \cdot g} \quad \text{or} \quad e = \frac{(b + \sqrt{bj})}{b-j} \cdot g.$$

Rewriting the two solutions as

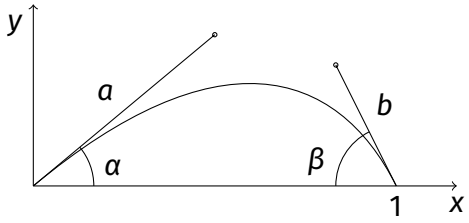
$$e = \frac{\sqrt{b}}{\sqrt{b} + \sqrt{j}} \cdot g \quad \text{or} \quad e = \frac{\sqrt{b}}{\sqrt{b} - \sqrt{j}} \cdot g$$

shows that the first solution in contrast to the second solution lies in the interval between 0 and g, and therefore is the desired solution.

Unfortunately, the formula does not work for more than two G^1–continuous quadratic Bézier curves. However, iteration seems to be stable (not proved yet).

6.3 G^2–continuity For Cubic Bézier Curves By Scaling Handles

Given a cubic Bézier curve $\binom{x}{y} = 3t(1-t)^2\binom{a\cos(\alpha)}{a\sin(\alpha)} + 3t^2(1-t)\binom{1-b\cos(\alpha)}{b\sin(\alpha)} + t^3\binom{1}{0}$ with $a, b \geq 0$



we calculate the curvature at $t = 0$

$$\kappa_\alpha = \kappa(0) = \begin{cases} \frac{2}{3a^2}\big(b\sin(\alpha+\beta) - \sin(\alpha)\big) & \text{if } a \neq 0, \\ \pm\infty & \text{else.} \end{cases}$$

and $t = 1$:

$$\kappa_\beta = \kappa(1) = \begin{cases} \frac{2}{3b^2}\big(a\sin(\alpha+\beta) - \sin(\beta)\big) & \text{if } b \neq 0, \\ \pm\infty & \text{else.} \end{cases}$$

Solving the equation system

$$\kappa_\alpha = \frac{2}{3a^2}\big(b\sin(\alpha+\beta)-\sin(\alpha)\big) \quad \text{and} \quad \kappa_\beta = \frac{2}{3b^2}\big(a\sin(\alpha+\beta)-\sin(\beta)\big)$$

for a and b is done by solving the (depressed) quartic equation

$$0 = 27\kappa_\alpha\kappa_\beta^2\cdot x^4 + 36\kappa_\alpha\sin(\beta)\kappa_\beta\cdot x^2 - 8\sin(\beta)a^3\cdot x + 8\sin(\alpha)\sin(\beta)a^2 + 12\kappa_\alpha\sin(\beta)^2$$

for b and then $a = \frac{2\sin(\beta)+3\kappa_\beta b^2}{2\sin(\alpha+\beta)}$ if $\alpha+\beta \neq 0$. If $\alpha+\beta = 0$, the upper equation system simplifies to

$$\kappa_\alpha = -\frac{2\sin(\alpha)}{3a^2} \quad \text{and} \quad \kappa_\beta = \frac{2\sin(\alpha)}{3b^2}$$

with the non–negative solutions

$$a = \sqrt{-\frac{2\sin(\alpha)}{3\kappa_\alpha}} \quad \text{and} \quad b = \sqrt{\frac{2\sin(\alpha)}{3\kappa_\beta}}$$

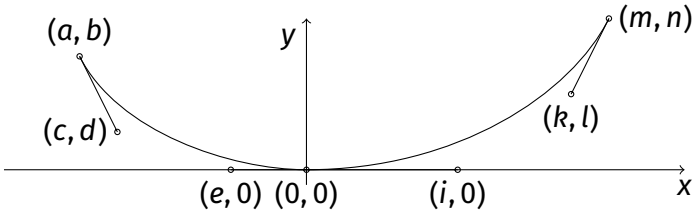
as long as $\kappa_\alpha \neq 0 \neq \kappa_\beta$ (otherwise there is no solution). Now we can take the average curvature between segments and adjust the handles as described above and the curves will join G^2–continuous.

In fact, not every average curvature will produce useable handles on both sides. Therefore, iteratively taking the average curvature is recommended.

If we have a sharp end at α (i.e. a non–smooth node) it is additionally convenient to choose a such that $\max|\kappa(t)|$ is minimal. In practice this can be done by varying b and calculating $a = \frac{2\sin(\beta)+3\kappa_\beta b^2}{2\sin(\alpha+\beta)}$ and $\max|\kappa(t)|$ for each «sensible» $b \geq 0$ (b should not exceed the tangent triangle). If $\alpha+\beta = 0$, b is fixed and a can be varyied independently of b.

7 G^3-Continuity For Two Adjacent Cubic Bézier Curves

We consider two G^1–continuous cubic Bézier curves $\binom{x}{y} = (1-t)^3\binom{a}{b} + 3t(1-t)^2\binom{c}{d} + 3t^2(1-t)\binom{e}{0}$ and $\binom{x}{y} = 3t(1-t)^2\binom{i}{0} + 3t^2(1-t)\binom{k}{l} + t^3\binom{m}{n}$ with $e < 0 < i$:



For the G^3–continuity at their joint $(0,0)$ we have to equalize the curvature

$$\kappa(t) = \frac{x'(t)\cdot y''(t) - x''(t)\cdot y'(t)}{\big(x'(t)^2 + y'(t)^2\big)^{\frac{3}{2}}}$$

on both sides and the derivate of the curvature on both sides:

$$\frac{2l}{3i|i|} = -\frac{2d}{3e|e|} \quad \text{and} \quad \frac{18il + 2in - 12kl}{3i^2|i|} = \frac{18ed + 2eb - 12cd}{3e^2|e|}$$

Because of $e < 0 < i$ this simplifies to

$$\frac{l}{i^2} = \frac{d}{e^2} \quad \text{and} \quad \frac{9il + in - 6kl}{i^3} = -\frac{9ed + eb - 6cd}{e^3}.$$

Solving these two equations for e and i yields to two solutions

$$\boxed{e = \frac{6d(cl - k\sqrt{dl})}{dn + 18dl + bl} \quad \text{and} \quad i = \frac{6l(dk - c\sqrt{dl})}{dn + 18dl + bl}} \quad \text{or} \quad e = \frac{6d(cl + k\sqrt{dl})}{dn + 18dl + bl} \quad \text{and} \quad i = \frac{6l(dk + c\sqrt{dl})}{dn + 18dl + bl}.$$

If we assume $b > 0$, $c < 0$, $d \geq 0$, $k > 0$, $l \geq 0$ and $n > 0$ (which is the generic case in typedesign) the second solution can lead to cases where $e > 0$ or $i < 0$, whereas the first solution always lead to $e \leq 0 \leq i$. Therefore, the algorithm in *Curvatura* uses the first solution.

Note:

- $e = 0$ if $d = 0$
- $i = 0$ if $l = 0$

Unfortunately, the formula does not work for more than two G^1–continuous cubic Bézier curves (but iteration is stable in most of the cases).

The big problem of this algorithm is that after applying it, the handles may exceed the tangent triangle, which means that an additional inflection point occurs on the segment. Therefore, this algorithm has been removed from *Curvatura*.