Numerical solutions for G^2 Hermite interpolation problem with spirals

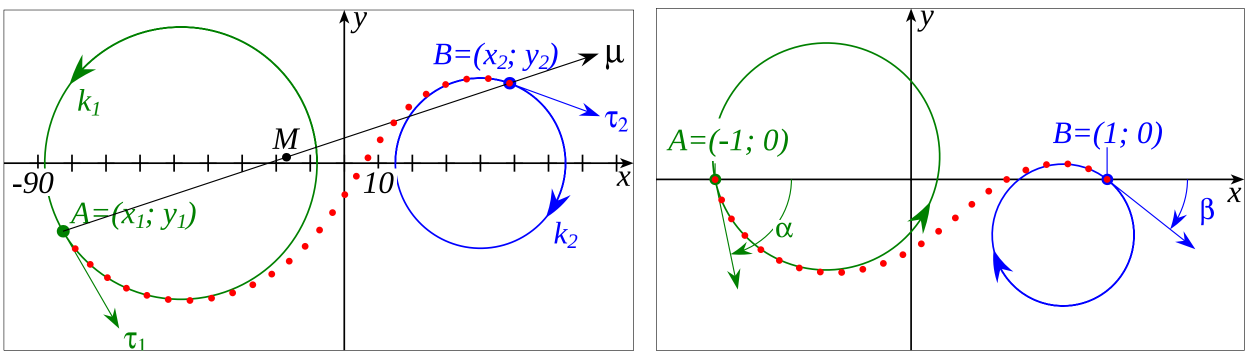
A. K.

October 15, 2019

This project is intended to provide a numerical solution (or several solutions) of the **two-point G^2 Hermite interpolation problem with spirals**.
I. e., a *transition curve*, joining two given points A and B, is constructed, matching given tangents and curvatures at A and B (Figure 1). *Spirality* means the monotonicity of the Chesaro equation of the transition curve: function

$$k(s) \equiv \tau_s'(s)$$

is monotonous (k being curvature, s arc length, and τ the direction of the tangent to the curve).



VogtData

Figure 1. Example of 2-point G^2 Hermite data (boundary conditions).

The left picture in Figure 1 shows the example of 2-point G^2 Hermite data as:
point $A = (x_1, y_1)$, tangent τ_1 and curvature k_1 at A,
point $B = (x_2, y_2)$, tangent τ_2 and curvature k_2 at B. Namely,

$$\begin{aligned} &x_1 = -82.64, \quad &y_1 = -20., \quad &\tau_1 = 300.0^\circ, \quad &k_1 = 0.025; \\ &x_2 = 48.55, \quad &y_2 = 23.5, \quad &\tau_2 = -20.0^\circ, \quad &k_2 - 0.04; \end{aligned}$$

Dotted curve is a an example of a spiral, matching this data. Its curvature is decreasing ($k_1 > k_2$), and has an inflection (due to $k_1 k_2 < 0$).

Normalization of given G^2 data. The right picture in Figure 1 shows the same data brought to *normalized position*, i. e. to the local coordinate system, aligned with the chord. Denote $2c$ the chord length, and μ its direction:

$$c = \frac{1}{2}\,|AB|\,, \quad \mu = \arg\left[x_2 - x_1 + \mathrm{i}(y_2 - y_1)\right] = \mathtt{atan2}(y_2 - y_1,\, x_2 - x_1).$$

Normalization involves:

- moving point $M = (A+B)/2$ to the coordinate origin;
- rotating by the angle $-\mu$ to align the chord $\overrightarrow{AB}$ with the x-axis;
- scaling (homothety) by the factor c^{-1} such that $A \to (-1;\,0)$, $B \to (1;\,0)$.

In this coordinate system the boundary conditions are transformed to:

$$\begin{aligned} (x_1, y_1) &\to (-1; 0), \qquad \tau_1 \to \alpha = \tau_1 - \mu, \qquad k_1 \to a = k_1 c; \\ (x_2, y_2) &\to (+1; 0), \qquad \tau_2 \to \beta = \tau_2 - \mu, \qquad k_2 \to b = k_2 c. \end{aligned}$$

The program shows given data and solutions in both original (Page 1) and normalized coordinate systems (Page 2 and pages for every solution found). Additional transformation is used in calculations, transforming the case of decreasing curvature into increasing one by the the symmetry about the x-axis.

Existence of solutions. The above transformations do no affect the values of the invariants Q (1) and $\sigma = \alpha + \beta$ (invariance of σ may require corrections $\alpha \to \alpha \pm 2\pi$, $\beta \to \beta \pm 2\pi$).

- A spiral arc (non-biarc), matching given 2-point G^2 data exists iff

$$Q = (k_1 c + \sin\alpha)(k_2 c - \sin\beta) + \sin^2 \frac{\alpha+\beta}{2} < 0. \tag{1}$$

DefQ

- A *short* spiral is a spiral arc which does not turn near endpoints. A *short* spiral arc exists if, additionally,

$$\begin{aligned} &\text{if } \ k_1 < k_2: \qquad -\pi < \alpha, \beta \leqslant \pi, \qquad \alpha + \beta > 0; \\ &\text{if } \ k_1 > k_2: \qquad -\pi \leqslant \alpha, \beta < \pi, \qquad \alpha + \beta < 0 \end{aligned} \tag{2}$$

Shortness

  $[\operatorname{sgn}(\alpha+\beta) = \operatorname{sgn}(k_2 - k_1) \neq 0]$.

- This method finds solution(s) if

$$0 < |\sigma| \leqslant \pi. \tag{3}$$

SigmaPi

  This includes all convex spirals ($k_1 k_2 \geqslant 0$) and some spirals with inflection (e. g., one-to-one projectable on the chord).

- A *long* spiral arc will be constructed if...

In the below description some family parameter Φ (`Phi`), $0 \leqslant |\Phi| \leqslant \Phi_{max} \leqslant 90°$, is introduced to parametrize the family of solutions. The above conditions being satisfied, at least one solution [1], that for $\Phi = 0$, always exists. It is the unique solution in the critical case $|\sigma| = \pi$.[1]

Brief description of the method ...see [1, 2] for details.

[1] Another method will be possibly added to this script (or the special script will be added) which solves the problem for any α, β, k_1, k_2 and multiple winding, provided that the necessary condition (1) holds.

1. Running the program

The program is written in PostScrpit language. You need some PostScript (GhostScript) interpreter to be inslalled. Under Linux it is usually `gv` or `gs`. Under Windows it is `gsview` or console application `gswin32c`.

You have to edit a few lines in the file `G2spiral.ps` to include your own boundary conditions (user data), and to run the program as

```
gv G2spiral.ps
```

Example of user input to construct the spiral in Figure 1 (red dotted curve):

```
<< /UserG2Data [/XYTK8 -82.64 -20. 300  0.025  48.55 23.5 -20.0 -0.04 ] >>
```

You can store user data in a separate file, say, `mydata.ps`, and run the program as

```
gv     -arg="-sFname=/home/ak/G2spiral/mydata.ps" G2spiral.ps
gs -dNOSAFER -sFname=/home/ak/G2spiral/mydata.ps  G2spiral.ps
```

(comments to NOSAFER option. . .)

Conversion to pdf by ImageMagick `convert` utility (requires GhostScript to be preinstalled):

```
convert G2spiral.ps G2spiral.pdf
```

(see `doc/Example-Converted_to_pdf.pdf`).

2. Setting user data

A few comments on PS syntax. Only data types, used in setting user data, are briefly commented below.

integer, real: as in all other languages: `1  1.  -3.14159`;

bool: `true  false`

string: string `"some text"` should be written in parenthesis as `(some text)`;

name: name starts with `/`, e. g. `/OutputFile` ;

array: `[el_0  el_1  el_2 ... ]`; array elements may be of different types;

dictionary: `<< /key1 value1   /key2 value2  /key3 value3 ... >>`.

Comment starts with `%`-sign:

```
% user data to approximate an arc of logarithmic spiral
% r(phi) = exp(phi*cot(nu)), 0 <= phi <= Phi
 [/LogSpiral2  80 180]          % nu = 80,  phi = 180
```

The angles are given in degrees.

Example of a file with user data. User data is stored as a dictionary like

```
<<
  /UserG2Data  [/XYTK8 -82.64 -20. 300  0.025  48.55 23.5 -20.0 -0.04 ]
  /UserPhiData [...]
  /Margin      30
  /OutputFile  (D:/tmp/VogtSpiral.txt)    % ouputs the curve as X Y pairs
>>
```

Only the first key-value pair, `/UserG2Data [ array ]`, is obligatory. Its possible versions are described below. The 1-st element of the array is some `/method`, followed by 2–8 numeric arguments.

1. `[/XYTK8   x1  y1  tau1    k1      x2   y2  tau2   k2]`
   Direct setting of boundary conditions. 8 arguments are $x_1, y_1, \tau_1, k_1, \quad x_2, y_2, \tau_2, k_2$.

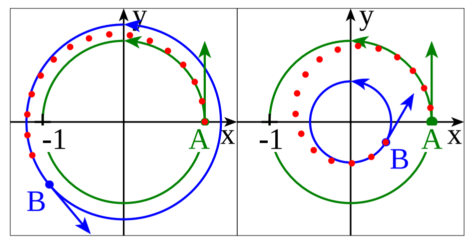
2. `[/Norm4 alpha k1   beta k2]`
   Data normalized to the unit chord:

   $$(x_1, y_1) = (-1, 0), \quad \tau_1 = \alpha, \quad k_1, \qquad (x_2, y_2) = (1, 0), \quad \tau_2 = \beta, \quad k_2.$$

   4 arguments, following the method identifier, are α, k_1, β, k_2.

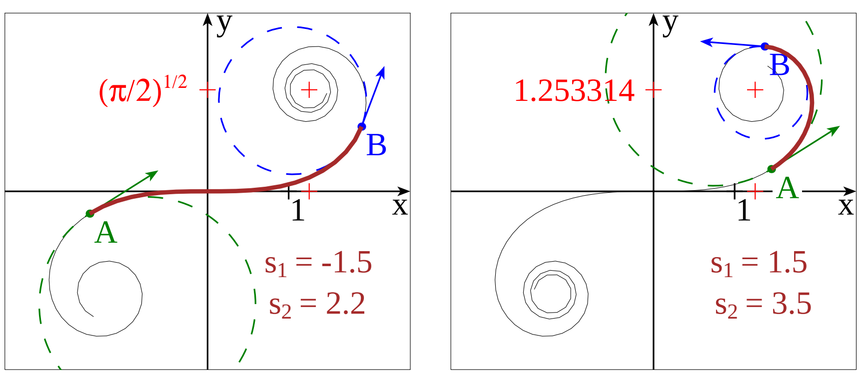
3. `[/Conc2 r2 phi2]`
   Concentric boundary conditions, $r_1 = 1$, $\varphi_1 = 0$, $\pi < \varphi_2 \leqslant 2\pi$, $r_2 \neq 1$:

   $$A = (x_1, y_1) = (1, 0), \qquad \tau_1 = \frac{\pi}{2}, \qquad k_1 = 1,$$
   $$B = (x_2, y_2) = \begin{pmatrix} r_2 \cos\varphi_2 \\ r_2 \sin\varphi_2 \end{pmatrix}, \quad \tau_2 = \varphi_2 + \frac{\pi}{2}, \qquad k_2 = \frac{1}{r_2}.$$



4. `[/Ell2 a b]`
   Boundary coonditions are those of a quarter of ellipse…

5. `[/Cornu2 s1 s2]`
   Boundary coonditions are borrowed from Cornu spiral with $k(s) = \dfrac{1}{2}s$, $s_1 \leqslant s \leqslant s_2$:



   Both the arc of the Cornu spiral and the approximating curves will be shown in the output pages.

6. `[/LogSpir2 nu phi]`
   Boundary conditions are borrowed from…

7. `[/ABpp4 alpha beta p1 p2]`
   (bilens parameters; for internal use).

The entry `/UserPhiData value` serves to select solutions from the whole family of solutions. If absent, default `/UserPhiData [0.]` is assumed. Possible values are:

`/UserPhiData N`, N integer. Tries to find solutions for $2N+1$ possible values of the family parameter Φ $(2N+3\,?)$. For $N=3$:
$\{-\Phi_{max}, -\Phi_2, -\Phi_1, \Phi_0 = 0., \Phi_1, \Phi_2, \Phi_{max}\}$.

`/UserPhiData [ ]` (empty array) – as the previous one with some automatic selection of N.

`/UserPhiData step`, step is real, in degrees (e. g., `5.` , not `5`). Tries to find solutions for $\Phi \in \{0., \pm\text{step}, \pm 2\cdot\text{step}, \pm 3\cdot\text{step}, \ldots, \pm\Phi_{max}\}$.

`/UserPhiData [Phi1 Phi2 Phi3 ... PhiN]` (reals) – list of desired family parameters. Note that such array can be set with PostScript `for`-loop as
`/UserPhiData [Phi1 step Phi2 {} for]`. E. g.,
`/UserPhiData [ -6.  1.5  3.1 {} for]` produces the array
`/UserPhiData [ -6. -4.5 -3. -1.5 0. 1.5 3.]`.

Other options are listed below (add comments!):

```
/Margin  real   % margins in mm
/Margin  int    % margins in pt: /Margin 72 corresponds to 1 inch.
```

Note that A4 paper is 210 x 297 mm, or 595 x 842 pt.

```
/OutputFile    string  %  e.g., (/home/ak/tmp/mySpiral_XY.txt)
/BlackWhite    false
/BaseLineSkip  20
/ShowAll       true   % Show page for every solution found
/MaxLength     1000. % Reject too long curves (usually applied to curves with inflecti
```

About PostScript error messages. . . .
Error messages due to PostScript syntax violation. . .
Errors due to forbidden output to disk. . .

Output. Output curve file is formatted as PostScript array for every solution found:
`[ x1 y1  x2 y2 ... xN yN ]`
Example:

```
% Phi: -45.0  Length: 4.08248
[
 -0.993767 -0.0829621 -0.987961 -0.0814893 -0.976391 -0.0786075 -0.953409 -0.0731016
 -0.908072 -0.0630687 -0.819742 -0.0465941 -0.651231 -0.0257036 -0.491357 -0.0172414
 .........................
 1.92118 0.795525 1.94762 0.903802 1.96055 1.00681 1.96188 1.10303
 1.95365 1.1915 1.93781 1.27174 1.91616 1.34367 1.86147 1.46366
 1.79837 1.55647
]
```

References

[1] *Kurnosenko A.I.* Two-point G^2 Hermite interpolation with spirals by inversion of hyperbola // Comp. Aided Geom. Design. 2010. V. 27. P. 474–481; ISSN 0167-8396, (http://www.sciencedirect.com/science/article/pii/S0167839610000270)

[2] *Kurnosenko A.I.* Two-point G2 Hermite interpolation with spirals by inversion of conics: summary // https://arxiv.org/abs/1401.7593