

# Numerical solutions for $G^2$ Hermite interpolation problem with spirals

A. K.

September 21, 2019

## Abstract

!!!

Wait for a few weeks for replacing this initial commit by something usable...

Library files are not yet included.

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., the *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 curve: function

$$k(s) \equiv \tau'_s(s)$$

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

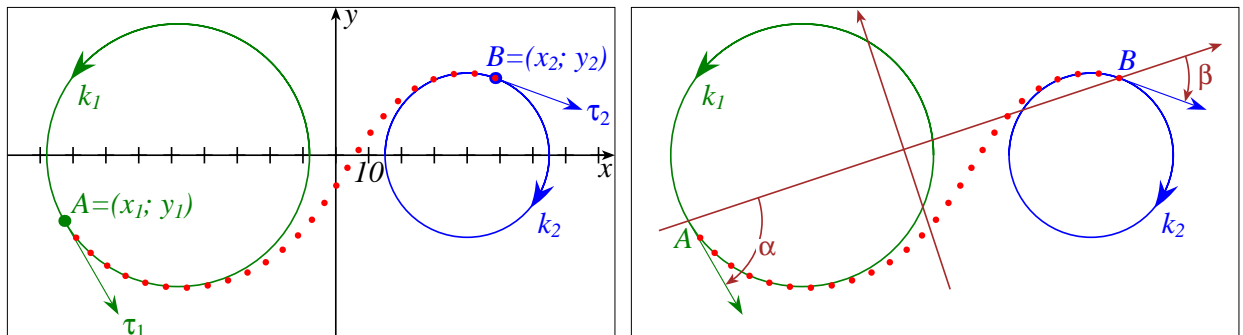


Figure 1. Example of 2-point  $G^2$  Hermite data (boundary conditions): point  $A = (x_1, y_1)$ , tangent  $\tau_1$  and curvature  $k_1$  at  $A$ , point  $B = (x_2, y_2)$ , tangent  $\tau_2$  and curvature  $k_2$  at  $B$ .

Existence of solutions. ...

Brief description of the method ... [1] (ref. to arxiv)

## 1. Running the program

The program is written in PostScript language. You need some PostScript (GhostScript) interpreter to be installed. 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 ] >>

      /method   x1    y1  tau1    k1      x2    y2  tau2    k2
```

You can also keep the user data in a separate file, say, `UserData.ps`, and run the program as

```
gv      -arg="-sFname=/home/ak/G2spiral/UserData.ps" G2spiral.ps
gs -dNOSAfer -sFname=/home/ak/G2spiral/UserData.ps  G2spiral.ps
(comments to NOSAFER ...)
```

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

```
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

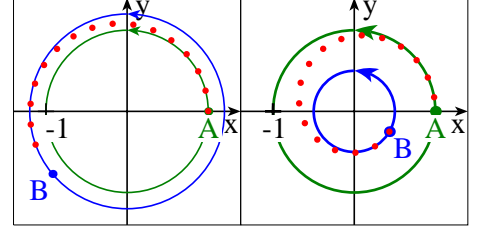
```
<<
  /UserG2Data [/XYTK8 -82.64 -20. 300 0.025 48.55 23.5 -20.0 -0.04 ]
  /UserPhiData [...]
  /Margin      30
  /OutputFile  (D:/tmp/VogtSpiral.txt)    % ouputs the curve as X Y pairs
>>
```

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  $x_1, y_1, \tau_1, k_1, x_2, y_2, \tau_2, k_2$ .
2. `/Norm4 alpha beta k1 k2]???` (check!)  
Data normalized to the unit chord  $(x_1, y_1) = (-1, 0)$ ,  $\tau_1 = \alpha$ ,  $(x_2, y_2) = (1, 0)$ ,  $\tau_2 = \beta$ .  
4 arguments, following the method, are  $\alpha, \beta, k_1, k_2$ . ???
3. `/Conc2 r2 phi2]`  
Concentric boundary conditions,  $r_1 = 1$ ,  $\varphi_1 = 0$ ,  $\pi < \varphi_2 \leq 2\pi$ ,  $r_2 \neq 1$ :

$$(x_1, y_1) = (1, 0), \quad \tau_1 = \frac{\pi}{2}, \quad k_1 = 1,$$

$$(x_2, y_2) = (r_2 \cos \varphi_2, r_2 \sin \varphi_2), \quad \tau_2 = \varphi_2 + \frac{\pi}{2}, \quad k_2 = \frac{1}{r_2}.$$



Add taus to figure.

4. `/El12 a b]`  
Boundary coonditions are those of...
5. `/Cornu3 s1 s2 f]`  
Boundary coonditions are borrowed from Cornu spiral with  $k(s) = \frac{s}{f^2}$ ?,  $s_1 \leq s \leq s_2$ ;  
 $f$  is the form parameter (scaling factor): distance ... Both the arc of the Cornu spiral  
and the approximating curve will be shown in the output. Add figure.
6. `/LogSpir2 nu phi2]`  
Boundary coonditions are borrowed from...
7. `/ABpp4 alpha beta p1 p2]`  
(bilens parameters; for internal use).

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

`/UserPhiData N`,  $N$  integer. Shows  $2N + 1$  solutions. 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  $N$ .  
This is the default behavior if this entry is absent in the user data dictionary.

`/UserPhiData step`,  $step$  is real, in degrees (e.g., `5.`, not `5`). Shows solutions for  
 $\Phi \in \{0., \pm step, \pm 2 \cdot step, \pm 3 \cdot step, \dots\}$ .

`/UserPhiData [Phi_min Phi_max N]`,  $N$  integer. Shows  $N$  solutions within the range  
 $[\Phi_{min}; \Phi_{max}]$ .

`/UserPhiData [Phi1 Phi2 Phi3 ... PhiN]` (reals) – list of desired family parameters.

**Other options** are listed below (add comments!):

```
/Margin real    % margins in mm                                A4 paper is 210. x
/Margin int     % margins in pt: /Margin 72 corresponds to 1 inch, A4 paper is 595 x 8
/OutputFile     string % e.g. (/home/ak/tmp/mySpiral_XY.txt)
/BlackWhite     false
/BaseLineSkip   20
/ShowAll        true
/LimitCrvLength 5000.
```

### About PostScript error messages. ...

Error messages due to PostScript syntax violation...

Errors due to absence of library files...

Errors due to forbidden output to disk...

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

```
% 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
-0.337668 -0.017637 -0.0404825 -0.0335358 0.25381 -0.0531306 0.402472 -0.0584971
0.552631 -0.0578547 0.703972 -0.0489771 0.855578 -0.029798 1.00592 0.00144225
1.15291 0.046022 1.29402 0.104561 1.42648 0.176884 1.54757 0.261951
1.65486 0.357907 1.74654 0.46223 1.82151 0.571974 1.87954 0.684055
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.