COMPUTING CMC AND SPHERICAL SURFACES BY THE DPW METHOD

DAVID BRANDER

This is an introduction on how to use Matlab to compute constant mean curvature surfaces and their parallel spherical surfaces using the DPW method [4]. An accessible introduction to DPW can be found in [5].

Spherical surfaces: Use the functions *K1surf* for a rectangular region and *K1surfpolar* for a disc around the origin.

CMC surfaces Can also be computed from the same functions (with H=1/2), or you can use the function *ecmch* and *ecmchpolar*.

The matlab functions can, at the time of writing, be found at: http://davidbrander.org/software.html.

1. CMC SURFACES

A surface is computed given a *potential*, which is the analogue for CMC surfaces of the Weierstrass data of minimal surfaces. Potentials for examples of non-minimal CMC surfaces that are deformations of minimal surfaces are readily supplied by using the formula (1.1) below, which comes from [2] (preprint). Solutions of Bj”orling's problem for non-minimal CMC surfaces (solved in [3]) can also easily be computed using the formulae in the appendix of [1].

1.1. Using the function ecmch.

The function ecmch computes the surface corresponding to the potential A. (A faster alternative to ecmch, that uses c++-complied mex functions is *ecmchX*). This is a loop valued function handle. The general form is described below in Section 1.4, but first we look at some examples.

One example of a potential is the matrix-valued function handle:

```
A=@(z,h)[0,-h,0,h-1,0,0; 0,0,1-h,0,h,0];
```

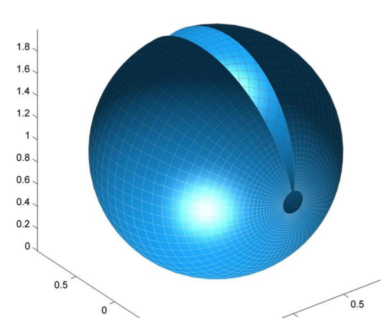
This is the boundary potential for the Björling problem where the initial curve is a circle and the prescribed normal along the curve is the curve's own normal (see Section 3.1 of [3]. The solution will be an unduloid, a cylinder, a sphere or a nodoid, depending on the value of h. The command:

```
f = ecmch(A, eye(2), [0, 0.05, 30, 30], [0, 0.05, 30, 30] , 3, 1);
```

produces a figure and some text output, a sample of which are displayed below:

```
Row 51 Max Error 1.8e-15. Errors: 6.3e-
Row 61 Max Error 2.5e-15. Errors: 6.2e-

Max error:3.9e-15. Mean error: 5.2e-16.
6.2e-16  6.2e-16
6.2e-16  6.2e-16
```

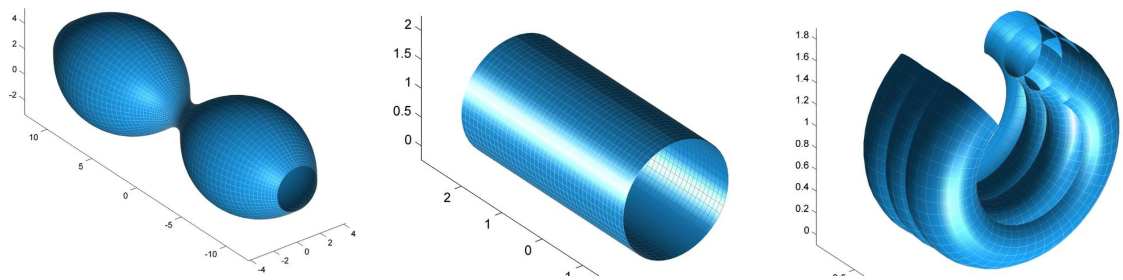


Date: June 5, 2015.

It is a sphere because we chose $h = 1$. The important figures: Max error:3.9e-15. Mean error: 5.2e-16, are near the bottom of the text output. The maximum error estimate is of the order 10^{-15}. These estimates are computed by checking that the matrix corresponding to the solution is in $\mathfrak{su}(2)$. Normally, if the maximum error is less than 10^{-1} and the mean error is less than 10^{-2} then the image is accurate, .i.e computing to higher accuracy will result in an image that is indistinguishable. To increase the accuracy, choose a higher order of polynomial approximation for the loops (3 was used here).

Similarly, choosing $h = 1/2$, $h = 2$ and $h = 0.2$, we get examples of the other types of surfaces of revolution (Figure 1.1), with:

```
f = ecmch(A, eye(2), [0, pi/50, 25, 25], [0, 0.055, 50, 50] , 5, 0.2);
f = ecmch(A, eye(2), [0, pi/50, 25, 25], [0, 0.05, 20, 20] , 4, 1/2);
f = ecmch(A, eye(2), [0, pi/50, 20, 20], [0, 0.05, 30, 30] , 6, 2);
```



$h = 0.2$ $h = 1/2$ $h = 2$

FIGURE 1. CMC surfaces of revolution.

Given a potential A, and an initial condition IC (usually the 2×2 identity matrix eye(2) in Matlab), to compute the corresponding CMC surface, enter:

```
f = ecmch(A, IC,  Ix , Iy, n, H);
```

The rectangle of integration is given by Ix and Iy. Ix is of the form *[x_0, stepsize, pointsleft, pointsright]*, representing the interval $[x_0 - pointsleft \times stepsize, x_0 + pointsright \times stepsize]$ and Iy is similar. The integration is done over the rectangle corresponding to $Ix \times Iy$, starting from the middle, the point (x_0, y_0). H is the mean curvature of the surface that will be computed, and can be any non-zero real number. The parameter n is the maximum order of polynomial approximation used. If the initial condition is the identity, the true solution has order 0 at the center point, and this grows higher as one moves away from this point. How quickly this grows depends on the potential A of course. The computation time increases with n. Depending on the problem, it is best to start with something like $n = 4$ and a 20×20 grid, (which takes about 1 second to integrate) and then modify this according to the output error estimates and image.

1.2. **Non-minimal surfaces associated to minimal surfaces.** Given the Weierstrass data (μ, ν) for a minimal surface,

$$f = 2\Re \int_{z_0}^{z} f_z \, \mathrm{d}z, \qquad f_z \, \mathrm{d}z = \big((1-\nu^2)e_1 - i(1+\nu^2)\,e_2 - 2\nu\, e_3\big)\, \mathrm{d}z,$$

with the coordinates chosen such that $\nu(z_0) = 0$ at some basepoint z_0, we show in [2] that a non-minimal CMC surface with the same Hopf differential is given by the potential with function handle:

$$\text{A= @(z,h) [0 ,-h } \mu(z)\text{, 0,0,0,0;} \quad \text{0,0, } -\tfrac{\partial \nu(z)}{\partial z}\text{,0,0,0].} \tag{1.1}$$

The potential is integrated with z_0 as the center point and the initial condition $IC =$ eye(2). More generally, if $\nu(z_0)$ is non-zero, one can instead use the potential

A= @(z,h) [0 ,-h $\mu(z)\Gamma_0(\bar{\nu}_0\nu+1)^2$, 0,0,0,0;   0,0, $-\frac{1}{\Gamma_0(\bar{\nu}_0\nu+1)^2}\frac{\partial\nu}{\partial z}$,0,0,0].

where

$$\Gamma_0 := \frac{\bar{\mu}(z_0)}{|\mu(z_0)|(|\nu(z_0)|^2+1)}.$$

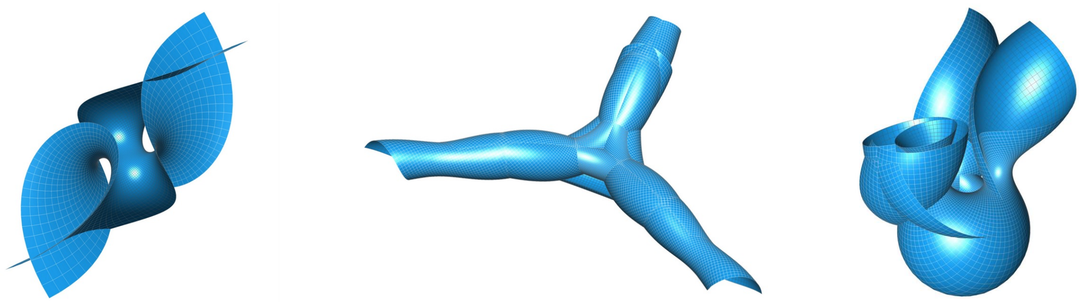
Example 1.1. *Enneper's surface:* Enneper's surface of order $k \geq 1$ is given by $\mu = 1$, $\nu = z^k$ on $\mathbb{C}$. So the non-minimal CMC h surfaces associated have potential:

A= @(z,h) [0 ,-h, 0,0,0,0;   0,0, -k*z∧(k-1),0,0,0].

For the case $k = 1$ and $h = 1$ we obtain a round cylinder. The other cases (for $H \neq 0$) are known as Smyth surfaces or $(k+1)$-legged Mister Bubbles.

We compute the case $k = 2$, for the values of $h = 0.000001$, 1 and 10:

```
A= @(z,h) [0 ,-h, 0,0,0,0; 0,0, -2*z, 0,0,0].
f = ecmch(A, eye(2), [0, 0.04, 30, 30], [0, 0.04, 30, 30] , 3,0.000001);
f = ecmch(A, eye(2), [0, 0.02, 80, 80], [0, 0.02, 80, 80] , 6,1);
f = ecmchX(A, eye(2), [0, 0.008, 60, 60], [0, 0.008, 60, 60] , 4,10);
```



$h = 0.000001$ $h = 1$ $h = 10$

FIGURE 2. Enneper surface of order 2 and 3-legged mister bubbles.

1.3. **Using *ecmchpolar*.** For surfaces with a point of symmetry, like Enneper's, a better image is obtained by using polar coordinates and computing a disc in the coordinate domain, using *ecmchpolar*. In polar coordinates, one replaces z with $r * exp(i * t)$, so the potential for the Enneper surface of order 2 is

```
A = @ (r,t,h)[0,-h,0,0,0,0;0,0,-2*r*exp(i*t),0,0,0];
```

The input data for ecmchpolar is of the form:

```
f = ecmchpolar(A, IC, Ir, It, n,H);
```

The integration is done in only one direction, radially from the middle, so Ir is expected to be of the form [0, stepsize, points], and It of the form [t_0, stepsize, points]. Loops are approximated to the same order n along each ray.

Given A as above, the following commands produce the images in Figure 3:

```
f=ecmchpolar(A, eye(2), [0,0.08,15], [0 pi/50, 100], 2,0.000001);
f=ecmchpolar(A, eye(2), [0,0.04,40], [0, pi/60, 120], 5,1);
f=ecmchpolar(A, eye(2), [0,0.012,40], [0, pi/100, 200], 3,10);
```

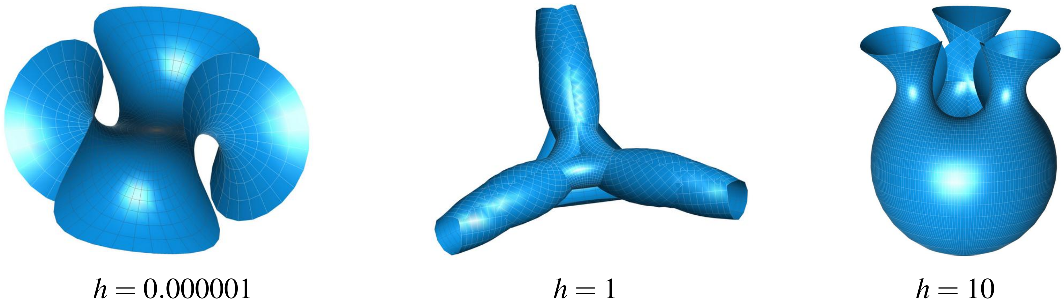


FIGURE 3. Enneper surface of order 2 and 3-legged mister bubbles computed on a polar region.

1.4. **The function handles for general DPW potentials.** To get a CMC-surface as output from ecmch, the function handle A has to have the appropriate properties for the DPW potential for a CMC surface [4]. The exact form, including the relationship with the Weierstrass data for minimal surfaces, can be found in [3] (see Theorem 2.6). Briefly, the twisted loop expression for A must be of the form:

$$A(z) = \sum_{n=-1}^{\infty} A_n(z)\lambda^n,$$

$$A_n(z) = \begin{pmatrix} d_n(z) & 0 \\ 0 & -d_n(z) \end{pmatrix}, \quad n \text{ even}, \qquad A_n(z) = \begin{pmatrix} 0 & a_n(z) \\ b_n(z) & 0 \end{pmatrix}, \quad n \text{ odd}, \tag{1.2}$$

where all functions are holomorphic. The surface is regular at points where $a_0(z) \neq 0$.

For ecmch, all loops are entered untwisted and as Laurent polynomials of the form $\sum_{-n}^{n} A_i \lambda^i$. (Coefficients of λ^{n+k}, for $k > 0$, are discarded). For example:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \lambda = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \lambda^{-1} + \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & b_{22} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \lambda$$

and this is entered as a 2×6 matrix

$$\begin{pmatrix} 0 & 0 & a_{11} & a_{12} & b_{11} & b_{12} \\ 0 & 0 & a_{21} & a_{22} & b_{21} & b_{22} \end{pmatrix} \quad \leftrightarrow [0,0,\ a_{11},a_{12},\ b_{11},b_{12};\quad 0,0,\ a_{21},a_{22},\ b_{21},b_{22}] \quad \text{in Matlab.}$$

In most of the literature on DPW, *twisted* loops are used, i.e. they satisfy the conditions at (1.2) on even and odd coefficients. Untwisting a twisted loop is done as follows:

$$\begin{pmatrix} a(\lambda) & b(\lambda) \\ c(\lambda) & d(\lambda) \end{pmatrix} \mapsto \begin{pmatrix} a(\sqrt{\lambda}) & B_{-1}(\sqrt{\lambda}) \\ C_{+1}(\sqrt{\lambda}) & d(\sqrt{\lambda}) \end{pmatrix}, \quad B_{-1}(\lambda) := \lambda^{-1} b(\lambda), \quad C_{+1}(\lambda) := \lambda c(\lambda).$$

For example if A is the twisted potential $\begin{pmatrix} 0 & -h\lambda^{-1} + (h-1)\lambda \\ (1-h)\lambda^{-1} + h\lambda & 0 \end{pmatrix} \mathrm{d}z$, the matrix untwists to

$$\begin{pmatrix} 0 & -h\lambda^{-1} + (h-1) \\ (1-h) + h\lambda & 0 \end{pmatrix} = \begin{pmatrix} 0 & -h \\ 0 & 0 \end{pmatrix} \lambda^{-1} + \begin{pmatrix} 0 & h-1 \\ 1-h & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ h & 0 \end{pmatrix} \lambda,$$

The potential would then be entered as: A=@(z,h)[0,-h,0,h-1,0,0; 0,0,1-h,0,h,0]; the potential for the Delaunay surfaces used above in Section 1.1.

2. Spherical Surfaces

This section describes how to compute spherical surfaces, as described in [1]. A lot of the details, for example the way a loop is represented as a function handle, are the same as for CMC surfaces, so in this section, we will mainly show how to use *K1surf* with examples. The mean curvature H is not included as a parameter in *K1surf*. It computes a CMC $1/2$ surface and the parallel spherical surface.

We use the functions *K1surfX.m, K1surfpolarX.m, loop2SMatC.mexw64, rloopEval.mexw64, gengcp.m, plotfp.m, sgcp.m, sing.m*. If these are not compatible with your system, then use *K1surf.m* and K1surfpolar.m instead, which should work on any system.

2.1. **Computing a surface from a normalized potential.** A normalized potential is of the form:
X=@(z)[0,a(z),0,0,0,0; 0,0,b(z) ,0,0,0],
where a and b are holomorphic functions. For example, the first image in Figure 4 is produced by the commands:

```
X=@(z)[0,1+z∧4,0,0,0,0; 0,0, z∧2,0,0,0];
[f,g] = K1surfX(X, eye(2), [0 0.02 40 40], [0 0.02 40 40], 5);
```

The surface f is plotted automatically. To plot the parallel CMC surface g, we can enter `plotfp(g);` and this produces the second image. We can compute the same surface on a polar disc with the commands:

```
Y=@(r,t)[0,1+(r*exp(1i*t))∧4,0,0,0,0; 0,0,(r*exp(1i*t))∧2,0,0,0];
[f,g]= K1surfpolarX(Y, eye(2), [0 0.02 45], [0 pi/100 200],4);
plotfp(g);
```

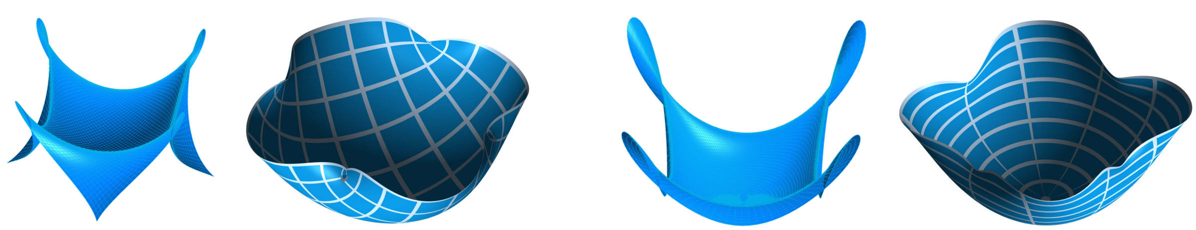


FIGURE 4. Symmetric spherical surface and the parallel CMC surface, computed on both rectangular and polar coordinate patches

2.2. **Solutions of the geometric Cauchy problem.** The function *gengcp.m* produces the potential for the regular geometric Cauchy problem (Theorem 4.4 in [1]). For example a non-orientable cylinder is computed as Example 4.5 in [1]. The data is $\kappa_n(s) = -\sin(s/2)$, $\kappa_g(s) = \cos(s/2)$ and $\mu(s) = 1/2$. We compute the solution with:

```
X=gengcp(@(t)-sin(t/2), @(t)cos(t/2), @(t)1/2);
[f,g] = K1surfX(X, eye(2), [0 pi/50 50 50], [0 0.04 30 30], 4);
```

This produces the first image in Figure 5.

We can create a plot showing the middle x coordinate line of the patch in red using *plotfp(f, numx, numy, middlelinewidth)*. This plots *numx* x-coordinate strips, *numy* y-coordinate strips and, if included it will plot a strip around the middle x coordinate line of width *middlelinewidth*. For example `plotfp(f,4,0,1);` produces the second image in Figure 5.

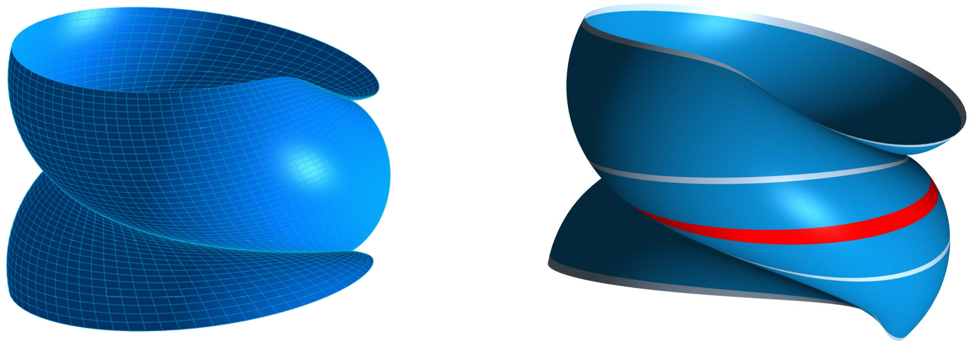


FIGURE 5. Non-orientable cylinder of constant Gauss curvature 1.

2.3. **The singular geometric Cauchy problem.** There are two types of potentials for producing surfaces with prescribed singular curves. Theorem 4.6 of [1] takes the curvature κ and the torsion τ of an arc-length parameterized curve and produces the spherical surface that contains this curve as a cuspidal edge. If the curvature vanishes to first order at a point where the torsion function is non-zero, we get a cuspidal beaks. The potential is produced by the function *sgcp.m*. For example a cuspidal beaks can be computed with:

```
X=sgcp(@(t)t, @(t)cos(t));
[f,g] = K1surfX(X, eye(2), [0 0.02 50 50], [0 0.02 50 50], 4);
```

And we can plot it showing a red band around the x-axis with the command plotfp(f,0,0,6). We need a wide band (here 6 steps in each y direction) for the strip to be visible, because the x parameter lines are very close together around a cuspidal edge. The result is displayed in Figure 6.

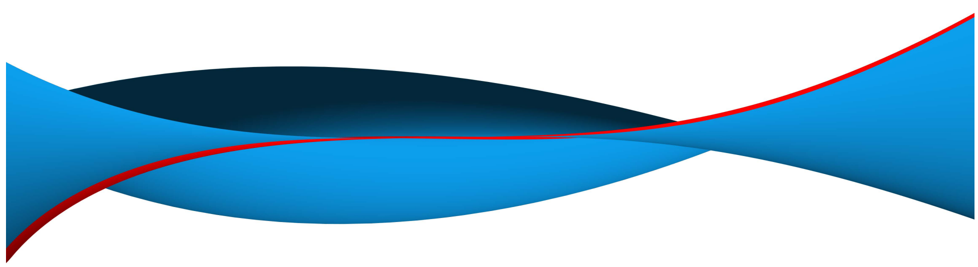


FIGURE 6. Cuspidal beaks singularity

To compute examples where the singular curve is non-degenerate but does not have a regular image in $\mathbb{R}^3$, such as a swallowtail, one can use Theorem 4.8 of [1]. The input is a pair of functions $b(t)$ and $c(t)$, where c is actually the geodesic curvature function for the curve in $\mathbb{S}^2$ corresponding to the normal of the solution surface along the curve $y = 0$. The non-degeneracy condition is $c \neq 0$. Given this, different choices of b give the following types of singularity at $(0,0)$:

(1) A cone point if $b \equiv 0$.
(2) Swallowtail if $b(0) = 0$ and $b'(0) \neq 0$.
(3) A cuspidal butterfly if $b(0) = b'(0) = 0$ and $b''(0) \neq 0$.
(4) A cuspidal edge if $b(0) \neq 0$.

The potential is produced by the function *sing.m*. An example with a swallowtail at $(k\pi, 0)$ for $k \in \mathbb{Z}$ is:

```
X=sing(@(t)sin(t), @(t)cos(t));
[f,g] = K1surfX(X, eye(2), [0 pi/100 110 110], [0 0.018 68 68], 5);
```

plotfp(f,10,20,5);

shown in Figure 7. It has degenerate singularities at $((2k+1)\pi/2, 0)$ for $k \in \mathbb{Z}$, where $c = 0$. These are apparently cuspidal beaks (although this question is not discussed in [1]).

FIGURE 7. Swallowtail and cuspidal beaks

REFERENCES

1. D Brander, *Spherical surfaces*, 2015, arXiv:1506.01605[math.DG].
2. D Brander and J Dorfmeister, *Deformations of constant mean curvature surfaces preserving symmetries and the Hopf differential*, Ann. Sc. Norm. Super. Pisa Cl. Sci. (5) **XIV** (2015), 1–31.
3. D Brander and J F Dorfmeister, *The Björling problem for non-minimal constant mean curvature surfaces*, Comm. Anal. Geom. **18** (2010), 171–194.
4. J Dorfmeister, F Pedit, and H Wu, *Weierstrass type representation of harmonic maps into symmetric spaces*, Comm. Anal. Geom. **6** (1998), 633–668.
5. S Fujimori, S-P Kobayashi, and W Rossman, *Loop group methods for constant mean curvature surfaces*, Rokko Lectures in Mathematics **17** (2005), arXiv:math/0602570 [math.DG].

DEPARTMENT OF APPLIED MATHEMATICS AND COMPUTER SCIENCE, MATEMATIKTORVET, BUILDING 303 B, TECHNICAL UNIVERSITY OF DENMARK, DK-2800 KGS. LYNGBY, DENMARK

E-mail address: `dbra@dtu.dk`