COMPUTING WILLMORE SURFACES WITH A NUMERICAL IMPLEMENTATION OF THE GENERALIZED BJÖRLING PROBLEM

DAVID BRANDER

This is a short note on how to compute Willmore surfaces with a numerical implementation of the DPW method, using a $5x5$ matrix representation via harmonic maps. The relevant theory is in [1] (Preprint Version). All the Matlab functions mentioned here can be found at http://davidbrander.org/software.html. Equivariant surfaces can easily be computed without reading about the theory. They come in families with several parameters, and the code here produces them automatically from the parameter values. The function *WdpwB* computes solutions from Björling-type potentials, or *boundary potentials*, which means potentials that are in the real-form of the loop group along the whole real line of the coordinate domain.

Algorithmically, the function is similar to the $2x2$ matrix implementations used for constant mean curvature surfaces. The $5x5$ underlying matrices slow this down. For a constant potential (*equivariant surfaces*), use the option $Isconstant = 1$ - this will cut computation time signficantly, usually to no more than one second. This makes it very easy to experiment with equivariant surfaces.

1. Computing equivariant surfaces

The potentials and initial conditions for all the different types of equivariant surfaces discussed in [1] are generated by the matlab functions *SOR, SOequi, hypequi_a1b0, hypequi_r1* and *hypequi_r1c0*. The meaning of the parameters is explained in [1].

1.1. **Computing a surface.**

We will compute a Willmore surface of revolution in $\mathbb{S}^3$. The potentials for these are given by the values of two real parameters (m,β) (see Theorem 5.3 of [1]). The function *WdpwB* is used by entering in Matlab's command line a command of the form:

$$[Ym, Yp] = WdpwB(X, IC, Ix, Iy, xmaxlooporder, ymaxlooporder, isconstant)$$

The type of values that the arguments should have are shown in the following example. We can compute the region with coordinate domain $[-\pi,\pi]\times[-\pi/2,\pi/2]$ of the SOR with parameter values $m = 1$ and $\beta = 1$, with the following two commands:

$$[X, IC] = SOR(1,1);$$

$$[Ym, Yp] = WdpwB(X, IC, [0, pi/100, 100, 100], [0, pi/100, 50, 50], 20, 30, 1);$$

Here X is the potential, IC the initial condition, which we got automatically by using the SOR function. $[0, pi/100, 100, 100]$ gives the initial point ($x_0 = 0$), the stepsize and the number of points to the left and right of x_0 to be computed for the x interval. $[0, pi/100, 50, 50]$ similarly represents the y-interval. So we have a rectangular grid. The next two parameters 20 and 30 tell the program how high to go with the polynomial approximations for loops, the first in the x-direction, the second in the y. The last parameter value, 1, tells the function that the potential is a *constant* function handle. Constant potentials are processed differently, saving time. Entering a 0 would give the same result (for constant potentials) but take maybe 100 times longer because the data is processed separately along each grid line in that case.

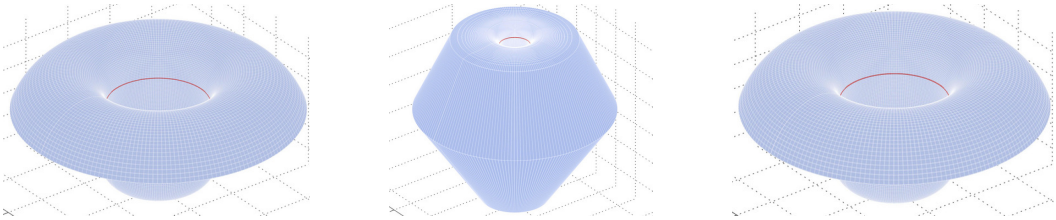

FIGURE 1. The same surface compute with loop orders respectively $(20,30)$, $(4,4)$ and $(4,8)$.

The result is the first image in Figure 1. It also outputs a list of numbers:
x errors: 6e-06 6e-06 5e-06 5e-06 5e-06 5e-06 5e-06 6e-06 6e-06 5e-06 5e-06 ...

Row 1. SO check: 0.059219, 0.00058771

y errors: 1e-06 7e-07 4e-07 4e-07 4e-07 4e-07 3e-07 2e-07 1e-07 2e-07 2e-07 1e-07 ...

Max x error: 6e-06, Max y Error=0.059219, Mean y Error=2.9239e-06, BC distance=0.

We only need to look at the last row of the numbers, in particular, the Max x error and Max y error. At each point, a number is calculated that measures how close the frame value is to being in the group $SO(1,4)$ (a very effective indicator for whether something went wrong). For equivariant surfaces, there are two sets of computations, one along the x-axis and one along the y-axis. In this case we see from the output *Max y Error=0.059219* that the Maximum error along the y-axs is 0.059, meaning a possible error on the order of 6 percent. The line immediately above shows all the errors along the y-axis. All the errors are insignificant except perhaps at one point:

2e-06 3e-06 6e-06 5e-05 0.06 5e-05 8e-06 4e-06 3e-06 3e-06 2e-06 9e-07

Isolated errors of this magnitude (about 0.1 or less), usually resulting from being close to the big cell boundary, have no visible effect on the image, and can be ignored. On the other hand, doing the same computation with a very low polynomial approximation order for the loops, say up to 4:

$$[Ym,Yp] = WdpwB(X,IC,[0,pi/100,100,100],[0,pi/100,50,50],4,4,1);$$

results in errors along the y-axis that look like this at one end:

y errors: 0.8 1 1 1 0.8 0.6 0.4 0.3 0.2 0.2 0.1

and like this at the other: 0.09 0.09 0.09 0.09 0.09 0.09 0.09 0.1 0.02 0.05

This produces the second image in Figure 1, which clearly has some errors in it. When the errors appear at either one or both ends of the interval, it normally means the order of approximation is too low for the size of the interval. You can either raise the loop order, or decrease the size of the interval. We raise the loop order along the y-interval: Doing the computation again with $xmaxlooporder = 4$ and $ymaxlooporder = 8$ produces errors like this:

y errors: 0.02 0.01 0.009 0.006 0.004 0.003 0.002 0.001 0.0008 0.0005 0.0003

and the third image in Figure 1, indistinguishable from the first image that had been computed with orders (20,30).

1.2. **Plotting different projections of the Willmore surface and its dual.** The command:

$$[Ym,Yp] = WdpwB(X,IC,[0,pi/100,100,100],[0,pi/100,50,50],10,10,1);$$

returns mesh data for two surfaces, Ym and Yp, where Ym is the solution Y for the Björling problem, as described in [1], and Yp is the dual Willmore surface, $\hat{Y}$. By default, the surface Ym in $\mathbb{S}^3$ is plotted, stereographically projected from the point $(0,0,0,1)$. To plot a projection of either $Y = Ym$ or $Y = Yp$, stereographically project from the point $\pm e_k$, (where $e_1 = (1,0,0,0)$, $e_2 = (0,1,0,0)$ etc), use the command *projplot(Y, ±k)*. For example, to plot the dual surface to the above surface, projected from the same point, use:

$$projplot(Yp,4);$$

The output of this and of the commands: $projplot(Yp,-4)$; $projplot(Yp,3)$; $projplot(Yp,-3)$; $projplot(Yp,2)$; are shown in Figure 2. One can also plot the stereographic projection with respect to an arbitrary point P by using the function *projplot2*

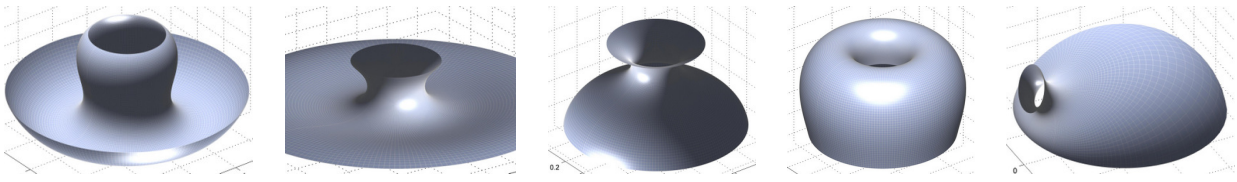

FIGURE 2. Five different projections of the surface Yp dual to the surface in Figure 1

thus: $projplot2(Ym,T)$, where T is a 4x4 orthogonal matrix that has P as its first row.

To just get the mesh-data in [X;Y;Z] matrix form for the surface, use the command *g=projplot(Y,k)*, which returns g=[X;Y;Z], the mesh-data of that projection.

1.3. **Other equivariant surfaces.** The other equivariant surfaces discussed in [1] are computed in a similar way, using the potentials and initial conditions obtained from the functions *SOequi, hypequi_a1b0, hypequi_r1* and *hypequi_r1c0*.

For example to compute one of the $SO(1,3)$-equivariant surfaces discussed in Section 6.1 (case a=1, b=0, c=1), use the function *hypequi_a1b0*. It has three parameters, h, r and θ, although θ is not relevant (up to conformal equivalence) and is optional. We compute the case $h=0$, $r=2$ with:

$$[X, IC] = hypequi\_a1b0(0,2);$$

$$[Ym, Yp] = WdpwB(X, IC, [0, pi/100, 150, 150], [0, pi/152, 100, 100], 20, 40, 1);$$

to get the first image in Figure 3. This is actually a minimal surface in $\mathbb{H}^3$ (at least the pieces inside and outside the unit sphere are) in the Poincare ball model. The projections with respect to the points $(0,0,-1,0)$ and $(-1,0,0,0)$, obtained by: $g = projplot(Ym,-3)$; and $g = projplot(Ym,-1)$; are also interesting, and are shown in the next two images in Figure 3. The second image is a cylinder, (or more like a twice punctured sphere geometrically).

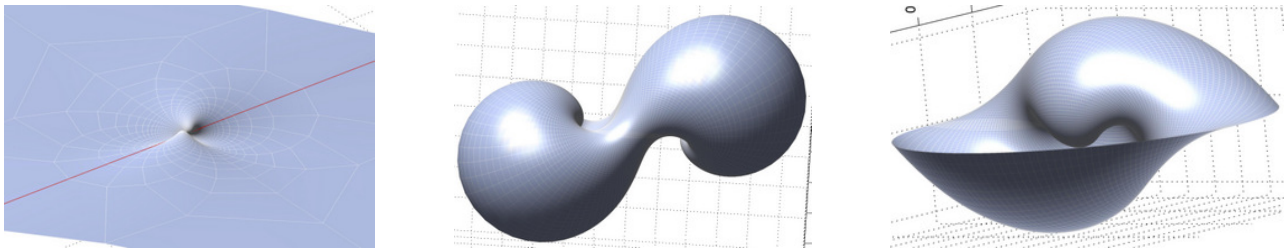

FIGURE 3. Three different projections of a minimal surface in hyperbolic 3-space

1.4. **Avoiding errors at the big cell boundary.** The Iwasawa decomposition for $SO(1,4)$, used in the DPW method implemented here, is only defined on a large open set, the big cell. When the boundary of this open set is approached, the $SO(1,4)$-frame, from which the surface is constructed, blows up. This has not been fully investigated yet for this problem, but it is likely that these are simply points where the Willmore surface Ym and its dual Yp coincide, which happens at umbilics. So, one expects the surface to be smooth there. In the numerical implementation, as long as one avoids these points there is no problem. The *complex* frame, which is the one used in the integration, does not blow up, so there is no loss of reliability for points on either side of the problem points. In other words, errors resulting from the big cell boundary do not spread to other parts of the surface.

The following example shows the simplest way to deal with errors arising in this manner, namely to shift the grid slightly to avoid the worst points. If we take the surface of revolution computed this way:

$$[X, IC] = SOR(1,8);$$

$$[Ym, Yp] = WdpwB(X, IC, [0, pi/100, 0, 20,], [0, pi/400, 0, 120], 26, 60, 1);$$

We get the following relevant errors, and the first image in Figure 4:

Max x error: 3e-08, Max y Error=0.99945, Mean y Error=0.00083639, BC distance=0

.... 0.001 0.003 0.02 0.8 1 0.05 0.01 0.001 0.003 0.0004 0.001 0.001 0.001 0.001 0.001 0.0002 4e-05 ....

The output "BC distance=0" means that the LU matrix decomposition used to approximate the Iwasawa splitting was, in fact, in the big cell, however one can guess that we were very close to the boundary by the large errors appearing just around one or two points. There is a lack of smoothness along one line in the image.

If we compute the same region over a finer mesh:

$$[Ym, Yp] = WdpwB(X, IC, [0, pi/100, 0, 20], [0, pi/800, 0, 240], 26, 60, 1);$$

we find that we do actually leave the big cell:

Max x error: 3e-08, Max y Error=0.99945, Mean y Error=0.00061417, BC distance=20

0.003 0.02 0.02 0.2 0.8 0.1 1 0.5 0.05 0.004 0.01 0.02 0.001

We can smooth out the image by choosing a coarser mesh, plotting the same region using 90 points instead of the original 120:

$$[Ym, Yp] = WdpwB(X, IC, [0, pi/100, 0, 20], [0, pi/300, 0, 90], 26, 50, 1);$$

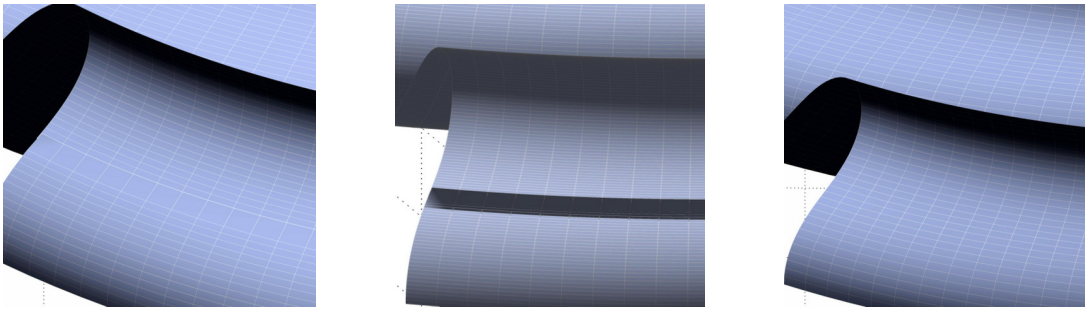

FIGURE 4. Avoiding errors at the big cell boundary

Max x error: 3e-08, Max y Error=0.92048, Mean y Error=0.00073605, BC distance=0
0.0004 0.001 0.002 0.003 0.01 0.3 0.9 0.05 0.007 0.005 0.0008

We still get an error of 0.9 at one point, which tells us that the matrix of the frame is not in $SO(1,4)$ at that point, but any error there might be in the surface Ym (obtained by subtracting the second column from the first and projecting to the sphere) is not visible in the image, the third image in Figure 4.

2. COMPUTING NON-EQUIVARIANT WILLMORE SURFACES USING THE BOUNDARY POTENTIALS

These are computed in much the same way as equivariant surfaces, only changing the value of the last parameter of *WdpwB* to 0 instead of 1. The computation time is much longer, and therefore it is not a good idea to experiment randomly with these.

To make the potential, use

$$X = bjorling(mu1, mu2, k1, k2, rho1, rho2);$$

where $mu1$, $mu2$, ... are real valued function handles for $\mu = mu1 + i\,mu2$, $k = k1 + ik2$, $\rho = rho1 + i\,rho2$ (see Theorem 4.2 of [1]).

Here is a somewhat random example generated by the command:

$$X = bjorling(@(t)cos(t), @(t)1, @(t)t, @(t)3 * sin(t), @(t)2 * t^2, @(t)3);$$

(The notation $@(t)cos(t)$ means the function $t \mapsto \cos(t)$ in Matlab). We compute a part of the corresponding solution to the Björling problem with:

$$[Ym, Yp] = WdpwB(X, eye(5), [0, pi/150, 40, 40], [0, pi/150, 40, 40], 15, 20, 0);$$

with the initial condition the 5x5 identity matrix *eye(5)* in Matlab, and obtain the first image in Figure 5. We can plot the dual surface, under the same projection, with

$$projplot(Yp, 4);$$

to get the second image in Figure 5.

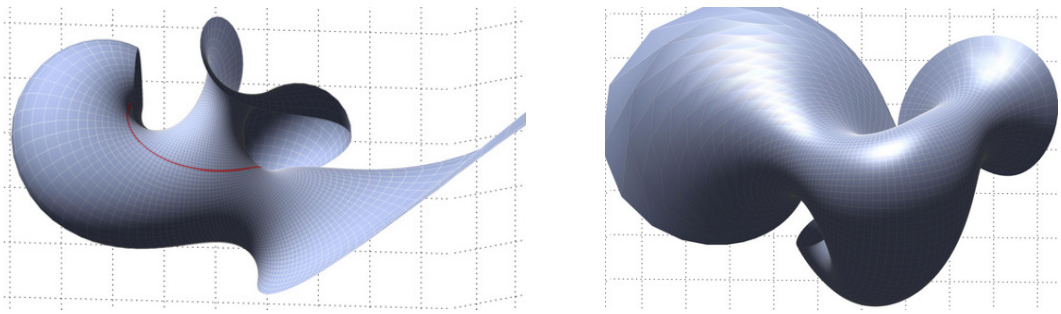

FIGURE 5. A "random" Willmore surface and its dual.

To compute solutions of the Björling problem with prescribed data along the curve, one needs to use the correct functions for μ, k and ρ, which can be obtained from equation (4.1) in [1]. For example, it is shown in Example 4.3 that a surface with the initial curves (lifted to $\mathbb{R}^5_1$)

$$Y_0(t) = (1, \cos(t), \sin(t), 0, 0), \qquad \hat{Y} = (1/2)(1, -\cos(t), -\sin(t), 0, 0),$$

and the initial sphere congruence:

$$\psi(t) = (0, 0, 0, -\sin(\theta(t)), \cos(\theta(t))),$$

have boundary potential data

$$(\mu, k, \rho) = (0, i\theta'(t)/2, -1/2).$$

We produce the potential for the case $\theta'(t) = \cos(t)$ with the command:

$$X = bjorling(@(t)0, @(t)0, @(t)0, @(t)cos(t)/2, @(t)(-1/2), @(t)0);$$

and compute it first on a thin strip around the x-axis and then on a larger domain:

$$[Ym, Yp] = WdpwB(X, eye(5), [0, pi/20, 20, 20], [0, pi/100, 10, 10], 10, 8, 0);$$

$$[Ym, Yp] = WdpwB(X, eye(5), [0, pi/30, 30, 30], [0, pi/50, 20, 20], 12, 16, 0);$$

The resulting surface is shown in Figure 6.

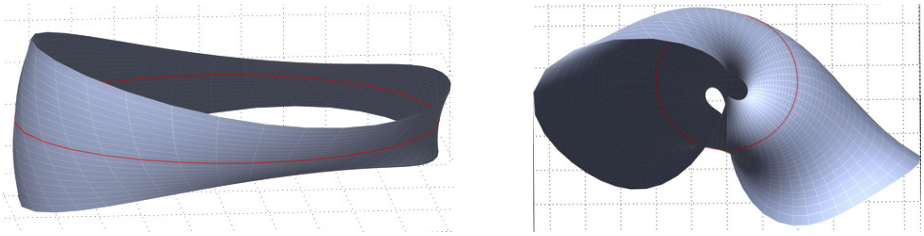

FIGURE 6. A Willmore surface containing a planar circle.

REFERENCES

[1] D Brander and P Wang, *On the Björling problem for Willmore surfaces*, Preprint (2014), arXiv:1409.3953.

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

E-mail address: D.Brander@mat.dtu.dk