Supplementary Material for "Structure–property relationships of cell clusters in biotissues: 2D analysis"

Xiaohua Zhou, Erhu Zhang, Minggang Xia, Lei Zhang, Zhiqi Tian, Jianlin Liu and Shengli Zhang

S.1. Details of the equilibrium shape equations without volume constraint

The shape equations for the S_1 part in $0 \le s \le s_0$ can be written as

$$\kappa_1 \ddot{\phi}_1 - \Delta p_1 x_1 \cos\phi_1 - \lambda_1 \sin\phi_1 = 0, \quad (S1)$$

$$\Delta p_1 x_1 \cos\phi_1 - \dot{\lambda}_1 = 0. \quad (S2)$$

For the S_3 part in $s_0 \le s \le s_0 + \frac{B}{2}$, there are

$$\kappa_3 \ddot{\phi}_3 - \Delta p_3 x_3 \cos\phi_3 - \lambda_3 \sin\phi_3 = 0, \quad (S3)$$

$$\Delta p_3 x_3 \cos\phi_3 - \dot{\lambda}_3 = 0. \quad (S4)$$

At the initial contact point $s = s_0$, the boundary conditions in Eqs. 4, 5 and 6 are changed to

$$\kappa_1 \dot{\phi}_1^2 + \kappa_2 \dot{\phi}_2^2 - \kappa_3 \dot{\phi}_{12}^2 = 2\omega, \quad (S5)$$

$$\kappa_1 \ddot{\phi}_1 + \kappa_2 \ddot{\phi}_2 - \kappa_3 \ddot{\phi}_3 = 0, \quad (S6)$$

$$\kappa_1 \dot{\phi}_1 + \kappa_2 \dot{\phi}_2 - \kappa_3 \dot{\phi}_3 = 0. \quad (S7)$$

Considering that all cells are equal, we have $\kappa_1 = \kappa_2 = \kappa_3/2 = \kappa$. Note that the S_1 and S_2 parts should be equal in a periodic system, and that the rotation directions of them are opposite in the adhesion region, at the point $s = s_0$, there are

$$\dot{\phi}_1 = -\dot{\phi}_2,\ \ddot{\phi}_1 = -\ddot{\phi}_2,\ \dot{\phi}_3 = \ddot{\phi}_3 = 0. \quad (S8)$$

The above conditions make sure that Eqs. 20 and 21 can be satisfied, and Eq. 19 is reduced to

$$\omega = \kappa \dot{\phi}_1^2, (s = s_0). \quad (S9)$$

Due to each adhesion part is a straight line, ϕ_3 should be independent to s and we choose $\phi_3 \equiv \phi_0 = \pi/n$. One can find that $n = 2$, 4 and 6 correspond to the shapes in Fig. 2(d), (e) and (f), respectively. In order to obtain dimensionless results, we choose $\kappa = 1$ and fix the girth of each cell $\mathcal{L} = \oint ds = 2\pi$.

First, we study the structures without volume constraint for each cell, which means the pressure difference $\Delta p_i = 0$. For convenient, in the following text we choose $\lambda_1 = \lambda$, $\phi_1 = \phi$, and $x_1 = x$. The shape equations for the S_1 part can be reduced to

$$\ddot{\phi} = \lambda \sin\phi, (0 \le s \le s_0). \quad (S10)$$

The first integral is

$$\dot{\phi}^2 = 2\lambda(C - \cos\phi), \quad (S11)$$

where C is an integral constant. Consequently, we have

$$ds = \frac{d\phi}{\sqrt{2\lambda(C - \cos\phi)}}. \quad (S12)$$

Choosing the initial conditions $x(0) = 0$, $y(0) = 0$, and $\phi(0) = 0$ and using the Eq. 7, the coordinates at the point $s = s_0$ are

$$x_0 = \int_0^{s_0} \cos\phi ds = \int_0^{\phi_0} \frac{\cos\phi d\phi}{\sqrt{2\lambda(C - \cos\phi)}}$$

$$= \sqrt{\frac{2}{\lambda(C-1)}} \times \left[C \times FN\left[\frac{\phi_0}{2}, \frac{2}{1-C}\right] - (C-1) \times SN\left[\frac{\phi_0}{2}, \frac{2}{1-C}\right]\right], \quad (S13)$$

$$y_0 = \int_0^{s_0} \sin\phi ds = \int_0^{\phi_0} \frac{\sin\phi d\phi}{\sqrt{2\lambda(C-\cos\phi)}} = \sqrt{\frac{2}{\lambda}}(\sqrt{C-\cos\phi_0} - \sqrt{C-1}), \quad (S14)$$

where $FN[x,y]$ and $SN[x,y]$ are the first and second incomplete elliptical integral, respectively. The length constraint $L = \oint ds = 2\pi$ gives the following equation

$$s_0 + \frac{B}{2} = \int_0^{\phi_0} \frac{d\phi}{\sqrt{2\lambda(C-\cos\phi)}} + \frac{B}{2} = \phi_0. \quad (S15)$$

It yields

$$\sqrt{\frac{2}{\lambda(C-1)}} \times FN\left[\frac{\phi_0}{2}, \frac{2}{1-C}\right] + \frac{B}{2} = \phi_0. \quad (S16)$$

Making use of Eq. S11, condition S9 is changed to

$$\lambda = \frac{\omega}{2(C-\cos\phi_0)}. \quad (S17)$$

Substituting the above equation into Eq. S16, we get

$$B = 2\phi_0 - 4\sqrt{\frac{C-\cos\phi_0}{\omega(C-1)}} \times FN\left[\frac{\phi_0}{2}, \frac{2}{1-C}\right]. \quad (S18)$$

The diameters of the inscribed circle and circumscribed circle of each cell in Fig. 2 are

$$R = 2x_0\csc\phi_0 + B\cot\phi_0, \quad (S19)$$

$$H = 2x_0\cot\phi_0 + B\csc\phi_0 + 2y_0, \quad (S20)$$

The dimensionless total energy for each cell is

$$E = \frac{2\pi}{\phi_0}\left(\int_0^{s_0} \frac{1}{2}\dot{\phi}^2 ds - \frac{1}{4}\omega B\right) = \frac{2\pi}{\phi_0}\int_0^{\phi_0} \frac{\lambda(C-\cos\phi)}{\sqrt{2\lambda(C-\cos\phi)}} d\phi - \frac{\pi}{2\phi_0}\omega B$$

$$= \frac{2\pi}{\phi_0}\sqrt{2\lambda(C-1)} \times SN\left[\frac{\phi_0}{2}, \frac{2}{1-C}\right] - \frac{\pi}{2\phi_0}\omega B. \quad (S21)$$

Substituting Eqs. S17 and S18 into the above equation, we obtain

$$E = \frac{2\pi}{\phi_0}\sqrt{\frac{\omega(C-1)}{C-\cos\phi_0}} \times SN\left[\frac{\phi_0}{2}, \frac{2}{1-C}\right] - \pi\omega + \frac{2\pi}{\phi_0}\sqrt{\frac{\omega(C-\cos\phi_0)}{C-1}} \times FN\left[\frac{\phi_0}{2}, \frac{2}{1-C}\right]. \quad (S22)$$

In the above equation, if ω and ϕ_0 are known, there is only an unknown constant C. Defining $\chi = 1/C$, the equilibrium shapes satisfy

$$\frac{dE}{dR} = \frac{dE/d\chi}{dR/d\chi} = 0. \quad (S23)$$

We find it needs $\chi \to 0$ ($C \to \pm\infty$). Then substituting Eq. S17 into Eq. S11, we have

$$\dot{\phi}^2 = \frac{\omega(C-\cos\phi)}{C-\cos\phi_0} = \omega, \quad (S24)$$

This result indicates that the optimal shapes of the free parts in Fig. 2 are circular arcs with the same radius $r_0 = \sqrt{1/\omega}$ and the total energy for each cell $E = 2\pi\sqrt{\omega} - \pi\omega$. Then the equilibrium distance between two cell is $R_0 = 2r_0 + B\cot\ \phi_0 = \ 2\sqrt{1/\omega} + 2(1-\sqrt{1/\omega})\ \phi_0\cot\phi_0$. $(S25)$

It is the result in Eq. 16.

S.2. Surface Evolver code for bead-like structure

Here we show a bead-like structure composed by three cells adhere together. This model needs to run on

the Surface Evolver 2.70s.

```
// Surface Evolver code for bead-like structure composed by three cells.
SPACE_DIMENSION 2
STRING

PARAMETER ad = -4               // ad is the adhesion potential
PARAMETER inputrv = 0.9    // inputrv is the reduced volume
VIEW_MATRIX
        1.000000000000000     0.000000000000000    -0.952000000000001
        0.000000000000000     1.000000000000000    -0.700000000000000
        0.000000000000000     0.000000000000000     1.000000000000000
define edge attribute sqcurve_string_mark integer
QUANTITY sqc_c1 ENERGY method sqcurve_string_marked global modulus 0.5 parameter_2 1
QUANTITY sqc_c2 ENERGY method sqcurve_string_marked global modulus 0.5 parameter_2 2
QUANTITY sqc_c3 ENERGY method sqcurve_string_marked global modulus 0.5 parameter_2 4
QUANTITY c1_length FIXED = 2*Pi method edge_length
QUANTITY c2_length FIXED = 2*Pi method edge_length
QUANTITY c3_length FIXED = 2*Pi method edge_length
vertices              /*    coordinates    */
1     0        0
2     1        0
3     0        1
4     1        1
5     0        2
6     1        2
7     0        3
8     1        3
edges
1     1     2    tension 0       sqcurve_string_mark 1       QUANTITY c1_length                                color blue
2     2     4    tension 0       sqcurve_string_mark 1       QUANTITY c1_length                                color blue
3     4     3    tension ad/2    sqcurve_string_mark 1+2    QUANTITY c1_length QUANTITY c2_length color red
4     3     1    tension 0        sqcurve_string_mark 1      QUANTITY c1_length                                color blue
5     4     6    tension 0        sqcurve_string_mark 2      QUANTITY c2_length                                color blue
6     6     5    tension ad/2    sqcurve_string_mark 2+4    QUANTITY c2_length QUANTITY c3_length color red
7     5     3    tension 0        sqcurve_string_mark 2      QUANTITY c2_length                                color blue
8     6     8    tension 0        sqcurve_string_mark 4      QUANTITY c3_length                                color blue
9     8     7    tension 0        sqcurve_string_mark 4      QUANTITY c3_length                                color blue
10    7     5    tension 0        sqcurve_string_mark 4      QUANTITY c3_length                                color blue
faces       /* edge loop */
1         1       2      3       4
2       -3       5      6       7
3       -6       8      9       10
bodies    /* facets */
1         1
2         2
3         3
read
setad:= {set edge tension ad/2 where color == 4 }
setrv:= { set body[1] target (inputrv*pi); set body[2] target (inputrv*pi); set body[3] target (inputrv*pi); }
suml := { La := sum(edges where color == 4,length);    La12 := sum(edges where original == 3,length);
             La23 := sum(edges where original == 6,length)}
report := { suml; rvt1 := body[1].volume/pi; rvt2 := body[2].volume/pi; rvt3 := body[3].volume/pi;
                  Ef := sqc_c1.value + sqc_c2.value + sqc_c3.value;      Et1 := sqc_c1.value + La12*ad/2;
                  Et2 := sqc_c2.value + La12*ad/2 + La23*ad/2;    Et3 := sqc_c3.value + La23*ad/2 ;
                  Ea:= ad*La; Et := Ef + ad*La;
                  printf "adhesion potential is %f, total adhesion length is %f, total energy is %f", ad, La, Et}
c1 := { { g 10; V } 100}
c2 := {r; r; r; {c1}10; r; {c1}10}
c3 := {r; r; r; {c1}10;setrv; {c1}10; r; {c1}10}
// Using the c3 or c2 order, we can obtain the structure with or without volume constraint.
```

S.3. Surface Evolver code for periodic square structure

Here we show a periodic square structure model which needs to run on the Surface Evolver 2.70s.

```
// Surface Evolver code for periodic square structure.
SPACE_DIMENSION 2
STRING
LINEAR
PARAMETER ad =    -4
PARAMETER RDx =    1.8    // period length on x direction
PARAMETER RDy =    1.8    // period length on y direction
PARAMETER inputrv = 0.94
PARAMETER RR = 1/4*(RDx+RDy)/cos(pi/8)

torus
periods
RDx    0
0    RDy

view_matrix
 1.0    0     3.65
 0     1.0    1.85
 0      0      1

define edge attribute sqcurve_string_mark integer
quantity sqc1 energy method sqcurve_string_marked modulus 0.5 global parameter_2 1
quantity sqc2 energy method sqcurve_string_marked modulus 0.5 global parameter_2 2
quantity sqc3 energy method sqcurve_string_marked modulus 0.5 global parameter_2 4
quantity sqc4 energy method sqcurve_string_marked modulus -0.5 global parameter_2 8
method_instance ves_length_1 method edge_length modulus 1
method_instance ves_length_2 method edge_length modulus 2
quantity ves_length fixed = 2*pi method ves_length_1 method ves_length_2

vertices
   1    RR*cos(-5*pi/24)-RDx          RR*sin(-5*pi/24)
   2    RR*cos(-pi/8)-RDx             RR*sin(-pi/8)
   3    RR*cos(pi/8)-RDx               RR*sin(pi/8)
   4    RR*cos(5*pi/24)-RDx           RR*sin(5*pi/24)
   5    RR*cos(19*pi/24)                RR*sin(19*pi/24)
   6    RR*cos(29*pi/24)                RR*sin(29*pi/24)
   7    RR*cos(7*pi/24)                 RR*sin(7*pi/24)-RDy
   8    RR*cos(3*pi/8)                  RR*sin(3*pi/8)-RDy
   9    RR*cos(5*pi/8)                  RR*sin(5*pi/8)-RDy
   10 RR*cos(17*pi/24)                   RR*sin(17*pi/24)-RDy
   11 RR*cos(31*pi/24)                   RR*sin(31*pi/24)
   12 RR*cos(41*pi/24)                   RR*sin(41*pi/24)

edges
   1       1    2     * *      tension 0            color blue       sqcurve_string_mark 1       ves_length_1
   2       2    3     * *      tension ad/2         color red        sqcurve_string_mark 1+2     ves_length_2
   3       3    4     * *      tension 0            color blue       sqcurve_string_mark 1       ves_length_1
   4       6    2     * *      tension 0            color blue       sqcurve_string_mark 2       ves_length_1
   5       3    5     * *      tension 0            color blue       sqcurve_string_mark 2       ves_length_1
   6       7    8     * *      tension 0            color blue       sqcurve_string_mark 4         ves_length_1
   7       8    9     * *      tension ad/2         color red        sqcurve_string_mark 2+4       ves_length_2
   8       9    10    * *      tension 0            color blue       sqcurve_string_mark 4         ves_length_1
   9       12   8     * *      tension 0            color blue       sqcurve_string_mark 2         ves_length_1
   10      9    11    * *      tension 0            color blue       sqcurve_string_mark 2         ves_length_1
   11      4    7     - +      tension 0            color blue       sqcurve_string_mark 1+4+8     ves_length_1
   12      5    10    * +      tension 0            color blue       sqcurve_string_mark 2+4+8     ves_length_1
   13      6    11    * *      tension 0            color blue       sqcurve_string_mark 2         ves_length_1
   14      12   1     + *      tension 0            color blue       sqcurve_string_mark 1+2+8     ves_length_1

faces       /* edge loop */
   1     1 2 3 11 6 7 8 -12 -5 -2 -4 13 -10 -7-9 14
```

```
bodies    /* facets */
   1            1

read
setrv := { set body[1] target (inputrv * pi)}
setad := { set edge tension ad/2 where color == 4}
suml := { La := 2*sum(edges where color == 4,length); Lf := sum(edges where color == 1,length)}
report := { suml;
   rvt := body[1].volume/pi;
   Ef := sqc1.value + sqc2.value+ sqc3.value+ sqc4.value;
   Ea := ad*La/2;
   Et := Ef + Ea;
   printf "ad is %f, La is %f, Lf is %f, Ef is %f, Et is %f, rv is %f", ad,    La, Lf, Ef, Et, rvt}
tran2 := { transform_expr "ab"; show_trans "R"; } // twice in each direction
tran3 := { transform_expr "abab"; show_trans "R"; } // three times in each direction
tran4 := { transform_expr "ababab"; show_trans "R"; } // four times in each direction
tran8 := { transform_expr "7(ab)"; show_trans "R"; } // four times in each direction
tran1 := { transform_expr ""; show_trans "R"; } // just the original
c1 := {{g 10; V }200}
c2 := {g;r;{c1}100;r;{c1}100;tran4}
c3 := {g;r;c1;r;setrv;{c1}100;{c1}100;tran4}
//Using the c3 or c2 order we can obtain periodic square structure with or without volume constraint.
//Using the c2 order, we can see that the adhesion regions are straight lines when there is not volume constraint.
```