free_shape_linear_fixed_h (Calls: 14324, Time: 18.120 s)

Generated 28-Dec-2022 15:12:05 using performance time.
Function in file C:\Users\zacky\OneDrive\Documents\MATLAB\Membranes_Blood\free_shape_linear_fixed_h.m
Copy to new window for comparing multiple runs

Parents (calling functions)

Function Name	Function Type	Calls
lin_global_search>stretch_bend_min	Subfunction	7160
lin_global_search>lipid_con_bend	Subfunction	7162
lin_global_search	Script	2

Lines that take the most time

Line Number	Code	Calls	Total Time (s)	% Time	Time Plot
110	`h = C(1)*log(r/lambda) + C(2) + C(3)*besselj(0,r/lamb...`	11042	5.102	28.2%	
112	`hderiv = C(1)./r-C(3)/lambda*besselj(1,r/lambda)-C(4)...`	11042	4.963	27.4%	
113	`lap_h = -C(3)/lambda^2*besselj(0,r/lambda)-C(4)/lambd...`	11042	4.548	25.1%	
65	`h = C(1)*log(r/lambda) + C(2) + C(3)*besseli(0,r/lamb...`	1830	0.484	2.7%	
67	`hderiv = C(1)./r+C(3)/lambda*besseli(1,r/lambda)-C(4)...`	1830	0.402	2.2%	
All other lines			2.621	14.5%	
Totals			18.120	100%	

Children (called functions)

Function Name	Function Type	Calls	Total Time (s)	% Time	Time Plot
trapz	Function	42972	0.830	4.6%	
Self time (built-ins, overhead, etc.)			17.290	95.4%	
Totals			18.120	100%	

Code Analyzer results

No Code Analyzer messages.

Coverage results

Show coverage for parent folder

Total lines in function	133
Non-code lines (comments, blank lines)	61
Code lines (lines that can run)	72
Code lines that did run	65
Code lines that did not run	7
Coverage (did run/can run)	90.28 %

Function listing

```
   Time    Calls  Line
                     1  function [h, C, A, E, lap h, hderiv] = free shape linear fixed h(r, r phi, d, phi, kappa, Sigma, h phi)
                     2  % solves the shape equation for the free surface of a section of membrane
                     3  % bound at one end to a sphere of radius R at the point phi radians from
                     4  % the bottom of the sphere, with a flat surface at d/2. The shape is
                     5  % characterised by the length lambda, which is equal to the square root of
                     6  % the membrane bending energy kappa divided by the tension Sigma. Also
                     7  % outputs the total area of the membrane, as well as the constants of
                     8  % integration.
                     9
  0.001    14324    10  if Sigma > 0
                    11
< 0.001     3282    12      lambda = sqrt(kappa/Sigma);
                    13
< 0.001     3282    14      if d/2/lambda>100
                    15
                    16          % asymptotic solution
  0.013     1452    17          C4 bar = (d/(2*r phi*lambda)*besseli(1,d/2/lambda,1)*h phi ...
            1452    18              - besseli(0,d/2/lambda,1)*tan(phi) -...
            1452    19              log(2*r phi/d)*d/(2*lambda)*besseli(1,d/2/lambda,1)*tan(phi))/...
```

```
         1452              (d/(2*r phi*lambda)*besselk(0,r phi/lambda,1)*besseli(1,d/2/lambda,1)...
         1452              + 1/lambda*besselk(1,r phi/lambda,1)*besseli(0,d/2/lambda,1)...
         1452              + log(2*r phi/d)*d/(2*lambda^2)*besselk(1,r phi/lambda,1)*besseli(1,d/2/lambda,1));
0.002    1452          C3 bar = (tan(phi)+1/lambda*besselk(1,r phi/lambda,1)*C4 bar)/...
         1452              (-d/(2*r phi*lambda)*besseli(1,d/2/lambda,1));

0.343    1452          h = -d/2/lambda*C3 bar*besseli(1,d/2/lambda,1)*log(r/lambda)...
         1452              +d/2/lambda*C3 bar*log(d/2/lambda)*besseli(1,d/2/lambda,1)...
         1452              -C3 bar*besseli(0,d/2/lambda,1)...
         1452              +exp((r-d/2)/lambda).*C3 bar.*besseli(0,r/lambda,1)...
         1452              +exp((r phi-r)/lambda).*C4 bar.*besselk(0,r/lambda,1);

0.281    1452          hderiv = -besseli(1,d/2/lambda,1)*C3 bar*d/(2*lambda)*1./r...
         1452              +exp((r-d/2)/lambda).*C3 bar/lambda.*besseli(1,r/lambda,1)...
         1452              -exp((r phi-r)/lambda).*C4 bar/lambda.*besselk(1,r/lambda,1);
0.273    1452          lap h = exp((r-d/2)/lambda).*C3 bar/lambda^2.*besseli(0,r/lambda,1)...
         1452              +exp((r phi-r)/lambda).*C4 bar/lambda^2.*besselk(0,r/lambda,1);

0.052    1452          A = 2*pi*trapz(r, ...
         1452              r.*sqrt(1+(hderiv).^2))...
         1452              + d^2*(1-pi/4);

                       % these are wrong! We actually don't want to solve them directly in
                       % the kappa -> 0 limit, since we need asymptotics
0.003    1452          C(1) = 0;
0.005    1452          C(2) = 0;
0.001    1452          C(3) = h phi/log(2*r phi/d);
< 0.001  1452          C(4) = -h phi*log(2*r phi/lambda)/log(d/2/lambda);

                       % Energy
0.054    1452          E = kappa/2*2*pi*trapz(r, r.*lap h.^2) + Sigma/2*2*pi*trapz(r, r.*hderiv.^2);

< 0.001  1830      else
0.005    1830          A c = d/(2*r phi*lambda)*besselk(1,d/2/lambda) - 1/lambda*besselk(1,r phi/lambda);
0.004    1830          B c = -d/(2*r phi*lambda)*besseli(1,d/2/lambda) + 1/lambda*besseli(1,r phi/lambda);
0.005    1830          D c = besseli(0,r phi/lambda) - besseli(0,d/2/lambda) ...
         1830              - log(2*r phi/d)*d/2/lambda*besseli(1,d/2/lambda);
0.004    1830          E c = besselk(0,r phi/lambda) - besselk(0,d/2/lambda) ...
         1830              + log(2*r phi/d)*d/2/lambda*besselk(1,d/2/lambda);

0.003    1830          C(4) = (h phi - D c*tan(phi)/B c)/(-A c*D c/B c+E c);
< 0.001  1830          C(3) = (tan(phi) - C(4)*A c)/B c;
0.003    1830          C(1) = -C(3)*d/2/lambda*besseli(1,d/2/lambda) + C(4)*d/2/lambda*besselk(1,d/2/lambda);
0.003    1830          C(2) = -C(1)*log(d/2/lambda) - C(3)*besseli(0,d/2/lambda) - C(4)*besselk(0, d/2/lambda);

0.484    1830          h = C(1)*log(r/lambda) + C(2) + C(3)*besseli(0,r/lambda) + C(4)*besselk(0,r/lambda);

0.402    1830          hderiv = C(1)./r+C(3)/lambda*besseli(1,r/lambda)-C(4)/lambda*besselk(1,r/lambda);
0.397    1830          lap h = +C(3)/lambda^2*besseli(0,r/lambda)+C(4)/lambda^2*besselk(0,r/lambda);

               %         area func = @(x) x.*sqrt(1+(C(1)./x+C(3)/lambda*besseli(1,x/lambda)-C(4)/lambda*besselk(1,x/lambda)
               %         bend func = @(x) x.*(C(3)/lambda^2*besseli(0,x/lambda)+C(4)/lambda^2*besselk(0,x/lambda)).^2;
               %         sig func = @(x) x.*(C(1)./x+C(3)/lambda*besseli(1,x/lambda)-C(4)/lambda*besselk(1,x/lambda)).^2;

0.062    1830          A = 2*pi*trapz(r, r.*sqrt(1+(hderiv).^2)) ...
         1830              + d^2*(1-pi/4);

               %         A = 2*pi*integral(area func,r phi,d/2) + d^2*(1-pi/4);

0.065    1830          E = kappa/2*2*pi*trapz(r, r.*lap h.^2) + Sigma/2*2*pi*trapz(r, r.*hderiv.^2);

               %         E = kappa/2*2*pi*integral(bend func,r phi,d/2) + Sigma/2*2*pi*integral(sig func,r phi,d/2);
< 0.001  3282      end

0.001    11042 elseif Sigma < 0

0.001    11042     lambda = sqrt(kappa/-Sigma);

                   % % fixed height
                   % M = [log(d/2), 1, besselj(0,d/2/lambda), bessely(0,d/2/lambda);...
                   %      2/d, 0, -besselj(1,d/2/lambda)/lambda, -bessely(1,d/2/lambda)/lambda;...
```

```
                    %      log(r phi/lambda), 1, besselj(0,r phi/lambda), bessely(0,r phi/lambda);...
                    %      1/r phi, 0, -besselj(1,r phi/lambda)/lambda, -bessely(1,r phi/lambda)/lambda];
                    %
                    % c = [0,0,h phi,tan(phi)]';
                    %
                    % C = M\c;

0.042     11042     H c = d/(2*r phi*lambda)*besselj(1,d/2/lambda) - 1/lambda*besselj(1,r phi/lambda);
0.037     11042     I c = d/(2*r phi*lambda)*bessely(1,d/2/lambda) - 1/lambda*bessely(1,r phi/lambda);
0.037     11042     F c = besselj(0,r phi/lambda) - besselj(0,d/2/lambda) ...
          11042         + log(2*r phi/d)*d/2/lambda*besselj(1,d/2/lambda);
0.042     11042     G c = bessely(0,r phi/lambda) - bessely(0,d/2/lambda) ...
          11042         + log(2*r phi/d)*d/2/lambda*bessely(1,d/2/lambda);

0.017     11042     C(4) = (F c*tan(phi)-H c*h phi)/(I c*F c-G c*H c);
< 0.001   11042     C(3) = (h phi - C(4)*G c)/F c;
0.024     11042     C(1) = C(3)*d/2/lambda*besselj(1,d/2/lambda) + C(4)*d/2/lambda*bessely(1,d/2/lambda);
0.023     11042     C(2) = -C(1)*log(d/2/lambda) - C(3)*besselj(0,d/2/lambda) - C(4)*bessely(0, d/2/lambda);

5.102     11042     h = C(1)*log(r/lambda) + C(2) + C(3)*besselj(0,r/lambda) + C(4)*bessely(0,r/lambda);

4.963     11042     hderiv = C(1)./r-C(3)/lambda*besselj(1,r/lambda)-C(4)/lambda*bessely(1,r/lambda);
4.548     11042     lap h = -C(3)/lambda^2*besselj(0,r/lambda)-C(4)/lambda^2*bessely(0,r/lambda);

                %     area func = @(x) x.*sqrt(1+(C(1)./x-C(3)/lambda*besselj(1,x/lambda)-C(4)/lambda*bessely(1,x/lambda)).^2

0.402     11042     A = 2*pi*trapz(r, r.*sqrt(1+(hderiv).^2)) + d^2*(1-pi/4);

                %     A = 2*pi*integral(area func,r phi,d/2) + d^2*(1-pi/4);

0.396     11042     E = kappa/2*2*pi*trapz(r, r.*lap h.^2) + Sigma/2*2*pi*trapz(r, r.*hderiv.^2);

                else

                    % if Sigma = 0, who knows what we get? I should figure out the limits!
                    h = zeros(size(r));
                    C = [0,0,0,0];
                    hderiv = zeros(size(r));
                    lap h = zeros(size(r));
                    A = d^2-pi*r phi^2;
                    E = 0;

0.003     14324 end
```

Local functions in this file are not included in this listing.