Constrained optimization using fmincon

Prepare workspace

```
clear
close all
fclose('all');
```

Scale our objective function to a unit cube

```
objFun = @(x) -1*computeFreq(x(1), x(2), pi*x(3));
```

Pick a random starting location

```
x0 = [rand() rand() rand()]
```

```
x0 = 1×3
    0.3674    0.9880    0.0377
```

Method choice

```
method = "Constraints+Bounds";
if strcmpi(method, "Constraints+Bounds")
    % Encode linear inequality constraints
    A = [ 1  0  0;...
          0  1  0;];
    b = [-0.9; -0.9;];

    % Encode additional constraints as bounds
    LB = [0.1 0.1 0];
    UB = [1 1 1];
elseif strcmpi(method, "Constraints-Only")
    A = [ 1  0  0;...
         -1  0  0;
          0  1  0;...
          0 -1  0];
    b = [-0.9; 0.1; -0.9; 0.1];

    % Basic bounds (can't have a "negative" radius)
    LB = [0.0 0.0 0];
    UB = [1 1 1];
end
```

No equality constraints

```
Aeq = [];
beq = [];
```

Specify function for non-linear constraints

```
nonlcon = @NonLinearConstraints;
```

Initialize log for output function

```
fid = fopen("optimLog.csv", "w+");
fprintf(fid, "x\ty\tz\tf\n");
fclose(fid);
```

Specify options for fmincon

```
options = optimoptions("fmincon");
options.Algorithm = "interior-point";
options.Display = "iter-detailed";
options.OutputFcn = @logFunction;
```

Solve problem using fmincon

```
sol = fmincon(objFun, x0, A, b, Aeq, beq, LB, UB, nonlcon, options);
```

```
                                            First-order      Norm of
 Iter F-count            f(x)  Feasibility   optimality         step
    0       4   -7.811458e+01    1.888e+00    2.131e+01
    1       8   -8.007949e+01    1.888e+00    1.557e+01    8.129e-02
    2      13   -7.999790e+01    1.887e+00    1.551e+01    1.667e-03
    3      17   -7.886912e+01    1.864e+00    1.543e+01    2.357e-02
    4      21   -7.454350e+01    1.608e+00    6.656e+01    2.687e-01
    5      25   -8.022150e+01    1.478e+00    7.052e+01    1.681e-01
    6      29   -8.124226e+01    1.454e+00    5.036e+01    2.437e-02
    7      34   -8.125312e+01    1.454e+00    4.625e+01    2.121e-04
    8      38   -8.126450e+01    1.453e+00    4.207e+01    2.222e-04
    9      42   -8.131773e+01    1.452e+00    4.234e+01    1.191e-03
   10      46   -8.144403e+01    1.449e+00    4.299e+01    2.798e-03
   11      50   -8.206765e+01    1.436e+00    4.603e+01    1.327e-02
   12      54   -9.342921e+01    1.271e+00    3.514e+01    1.678e-01
   13      58   -1.117653e+02    1.099e+00    1.747e+01    1.740e-01
   14      62   -1.118550e+02    1.098e+00    1.746e+01    7.244e-04
   15      66   -1.118550e+02    1.098e+00    1.746e+01    1.524e-08
   16      70   -1.118550e+02    1.098e+00    1.746e+01    3.026e-08
   17      74   -1.118550e+02    1.098e+00    1.746e+01    1.427e-07
   18      78   -1.118550e+02    1.098e+00    1.746e+01    7.109e-07
   19      82   -1.118550e+02    1.098e+00    1.746e+01    3.553e-06
   20      86   -1.118554e+02    1.098e+00    1.746e+01    1.777e-05
   21      90   -1.118569e+02    1.098e+00    1.746e+01    8.883e-05
   22      94   -1.118647e+02    1.098e+00    1.747e+01    4.442e-04
   23      98   -1.119038e+02    1.098e+00    1.749e+01    2.221e-03
   24     102   -1.121000e+02    1.098e+00    1.761e+01    1.111e-02
   25     106   -1.131189e+02    1.096e+00    1.870e+01    5.572e-02
   26     110   -1.211699e+02    1.083e+00    1.810e+01    2.952e-01
   27     114   -1.380999e+02    1.037e+00    3.788e+01    4.782e-01
   28     118   -1.390547e+02    1.071e+00    2.599e+01    3.445e-02
   29     122   -1.408422e+02    1.170e+00    2.486e+00    9.931e-02
   30     126   -1.407489e+02    1.178e+00    1.000e+00    8.410e-03

                                            First-order      Norm of
 Iter F-count            f(x)  Feasibility   optimality         step
   31     130   -1.407530e+02    1.179e+00    1.000e+00    5.181e-04
   32     134   -1.407530e+02    1.179e+00    1.000e+00    6.408e-06
   33     138   -1.407530e+02    1.179e+00    1.000e+00    4.388e-08
   34     142   -1.407530e+02    1.179e+00    1.000e+00    1.507e-08
   35     146   -1.407530e+02    1.179e+00    1.000e+00    2.356e-09
```

```
    36      150    -1.407530e+02    1.179e+00    1.000e+00    1.433e-08
    37      154    -1.407530e+02    1.179e+00    1.000e+00    1.210e-08
    38      158    -1.407530e+02    1.179e+00    1.000e+00    2.692e-08
    39      162    -1.407530e+02    1.179e+00    1.000e+00    1.647e-08
    40      166    -1.407530e+02    1.179e+00    1.000e+00    4.039e-08
    41      170    -1.407530e+02    1.179e+00    1.000e+00    2.285e-08
    42      174    -1.407530e+02    1.179e+00    1.000e+00    2.921e-07
    43      178    -1.407530e+02    1.179e+00    1.000e+00    2.825e-07

Optimization stopped because the relative changes in all elements of x are
less than options.StepTolerance = 1.000000e-10, but the relative maximum constraint
violation, 6.243375e-01, exceeds options.ConstraintTolerance = 1.000000e-06.
```

Unscale the solution vector

```
solVector = sol .* [1 1 pi];
```

Results

```
disp("r_1 = " + num2str(solVector(1)) + " || r_2 = " + num2str(solVector(2)) + " || theta = " +
```

```
r_1 = 0.1 || r_2 = 0.27874 || theta = 3.1357
```

Save the solution vector to a CSV file

```
fid = fopen("solution.csv", "w+");
fprintf(fid, "x\ty\tz\n");
fprintf(fid, "%12.8f \t %12.8f \t %12.8f \n",[solVector]);
fclose(fid);
```

Function definitions

Function evaluation (additive inverse is objective function)

```
function freq = computeFreq(r1,r2,theta)
    %surface fit for frequency on radial disk with 2 supports
    %r1 refers to the distance from center of the first support (0.1 - 0.9)
    %r2 is the distance from center to the second support (0.1 - 0.9)
    %theta is the angle between the supports (from center) (0 - pi)

    freq = (140.93-r1*25)+(-7.458*r1+9.1185)*theta+(-170)*r2+ ...
           (5.783*r1-10.367)*theta^2+(-8.1)*theta*r2+(117)*r2^2+ ...
           (4.2)*theta^3+(-28.075*r1+31.5125)*theta^2*r2+(15.63*r1-2.26)*theta*r2^2+ ...
           (-.7)*theta^4+(-.35)*theta^3*r2+(21.77*r1-27.862)*theta^2*r2^2;

end
```

Nonlinear constraints function

```
function [c,ceq] = NonLinearConstraints(x)
r1 = x(1);
r2 = x(2);
th = x(3);

c(1) = 0.1 - sqrt(r1^2 + r2^2 - 2*r1*r2*cos(th));  % Use law of cosines
ceq = [];
```

```
end
```

FMINCON output function

```
function stop = logFunction(x, optimValues, state)
if iscolumn(x)
    x = x';
end
fid = fopen("optimLog.csv", "a");
fprintf(fid, "%12.8f \t %12.8f \t %12.8f \t %12.8f \n",[x.*[1 1 pi] optimValues.fval]);
fclose(fid);
stop = false;
end
```