```
%{
Jacob Leonard
MATH 467 - Fall 2015
jaleonar@usc.edu
Revision History
Date                     Changes                     Programmer
-----------------------------------------------------------------
11/15/2015               Original                   Jacob Leonard
           ~Seeking Counsel from Professor Wang~
12/4/2015             Reverted Function             Jacob Leonard
12/4-12/2015           Troubleshooting              Jacob Leonard
12/13/2015      Added Backtracking Line Search      Jacob Leonard
12/15/2015       Analyzing Backtracking Values      Jacob Leonard
12/16/2015                Completed                 Jacob Leonard
%}
%}

%this script is for the conjugate gradient method with Fletcher-Reeves
%formula

%define the values of x and y from -2 to 2, increasing by 1/25, for 101
%values
for j = 1:101
    x(j) = (-2+(4*(j-1))/100);
    y(j) = (-2+(4*(j-1))/100);
end

%define an anonymous function handle for the equations that compose the gradient and the↙
hessian
f = @(x,y) ((x^4+y^4-6*x^2*y^2-1)^2+(4*x^3*y-4*x*y^3)^2);
G = {@(x,y) (8*x*(x^6+3*x^4*y^2+x^2*(3*y^4-1)+y^2*(y^4+3))),@(x,y) (8*y*(x^6+3*x^4*y^2+3*x^2*↙
(y^4+1)+y^2*(y^4-1)))};
%Gradient = [g{1}(x,y),g{2}(x,y)];
H = {@(x,y) (8*(7*x^6+15*x^4*y^2+x^2*(9*y^4-3)+y^2*(y^4+3))),@(x,y) (48*x*y*↙
(x^4+2*x^2*y^2+y^4+1));@(x,y) (48*x*y*(x^4+2*x^2*y^2+y^4+1)),@(x,y) (8*(x^6+9*x^4*y^2+3*x^2*↙
(5*y^4+1)+y^2*(7*y^4-3)))};
%Hessian = [H{1}(x,y),H{2}(x,y);H{3}(x,y),H{4}(x,y)];

%desired level of accuracy
tolerance = 10^(-7);

%this matrix defines the size of the final graph to be plotted for
%iterations
ConjugateSteps = zeros(101,101);
ConjugateValues = zeros(101,101);

%run the algorithm for conjugate gradient and fletcher reeves
for i = 1:101
    for j = 1:101
        Z(:,:,1) = [x(i);y(j)];
        %set the values in Z equal to 0 to track the progress
        Z(1,1,2:5000)=0;
        Z(2,1,2:5000)=0;
        g(:,:,1) = [G{1}(x(i),y(j)),G{2}(x(i),y(j))];
        %if the gradient is 0, the algorithm is considered converged
        if (g(1,1,1) == 0) && (g(1,2,1) == 0)
            ConjugateSteps(i,j) = 0;
            ConjugateValues(i,j) = f(Z(1,1,1),Z(2,1,1));↙
;
```

```
            continue
        end
        d(:,:,1) = -g(:,:,1);
        dT(:,:,1) = transpose(d(:,:,1));
        gT(:,:,1) = transpose(g(:,:,1));
        %backtracking values
        B = .8;
        A = .4;
        alpha1(1) = 1;
        for k = 2:5000
           %add backtracking to the line search
            s = 0;
            %return the backtracking values to 0
            alpha1(k) = 1;
            while s == 0
                %in order for the backtracking search to work, the
                %following rule must be satisfied: m>=c
                c = f(Z(1,1,k-1)+alpha1(k)*d(1,1,k-1),Z(2,1,k-1)+alpha1(k)*dT(2,1,k-1));
                m = f(Z(1,1,k-1),(Z(2,1,k-1)))+A*alpha1(k)*g(:,:,k-1)*dT(:,:,k-1);
                v = c-m;
                if v <= tolerance
                    alpha2 = alpha1(k);
                    break
                end
                alpha1(k) = B*alpha1(k);
            end
            Z(:,:,k) = Z(:,:,k-1)+(alpha2*dT(:,:,k-1));
            %if the function value of the algorithm dips below the desired
            %tolerance, then the algorithm is considered to have converged
            %to the optimal value
            if f(Z(1,1,k),Z(2,1,k))<tolerance
                ConjugateSteps(i,j) = k-1;
                ConjugateValues(i,j) = 0;
                break
            end
            %check to see if the Z values have NaN or Inf
            %values
            if (isnan(Z(1,1,k)) == 1) || (isnan(Z(2,1,k)) == 1)
                ConjugateSteps(i,j) = 5000;
                ConjugateValues(i,j) = 1;↙
;
                break
            end
            if (isinf(Z(1,1,k)) == 1) || (isinf(Z(2,1,k)) == 1)
                ConjugateSteps(i,j) = 5000;
                ConjugateValues(i,j) = 1;↙
;
                break
            end
            g(:,:,k) = [G{1}(Z(1,1,k),Z(2,1,k)),G{2}(Z(1,1,k),Z(2,1,k))];
            %if the gradient equals zero, then the algorithm is considered
            %to have converged
            if (g(1,1,k)) == 0 && (g(1,2,k) == 0)
                ConjugateSteps(i,j) = k-1;
                ConjugateValues(i,j) = f(Z(1,1,k),Z(2,1,k));↙
;
                break
            end
            %check to see if the gradient has NaN or Inf
```

```
            %values
            if (isnan(g(1,1,k)) == 1) || (isnan(g(1,2,k)) == 1)
                ConjugateSteps(i,j) = 5000;
                ConjugateValues(i,j) = 1;↙
;
                break
            end
            if (isinf(g(1,1,k)) == 1) || (isinf(g(1,2,k)) == 1)
                ConjugateSteps(i,j) = 5000;
                ConjugateValues(i,j) = 1;↙
;
                break
            end
            gT(:,:,k) = transpose(g(:,:,k));
            beta(k-1) = (g(:,:,k)*gT(:,:,k))/(g(:,:,k-1)*gT(:,:,k-1));
            d(:,:,k) = -gT(:,:,k)+(beta(k-1)*dT(:,:,k-1));
            dT(:,:,k) = transpose(d(:,:,k));
            %if the algorithm gets to the maximum number of steps without
            %satisfying any of the above criteria for convergence, or
            %divergence, then the max number of steps is recorded
            if k == 5000
                ConjugateSteps(i,j) = k;
                ConjugateValues(i,j) = f(Z(1,1,1),Z(2,1,1));
                break
            end
        end
    end
end

xAxis = linspace(-2,2,201);
yAxis = linspace(-2,2,201);

subplot(2,2,1:2)
%this plot will show the number of iterations it took
contourf(xAxis,yAxis,ConjugateSteps);
xlabel('x');
ylabel('y');
title('Conjugate Gradient Method with Fletcher-Reeves # of Steps');
colorbar;
subplot(2,2,3:4)
ConjugateValuesReal = real(ConjugateValues);
contourf(xAxis,yAxis,ConjugateValuesReal);
xlabel('x');
ylabel('y');
title('Binary Convergence Plot x=[-2:2], y=[-2:2]');
colorbar;
```