Descente de gradient à pas optimal

```
function [Xres] = desc_grad_opt(f, gradf, x0,
step, rel_tol, n_iterations, return_iterates)
%DESC_GRAD_OPT Summary of this
function goes here
%   Detailed explanation goes here
    X = x0 ;
    Xres = [X] ;
    n = 0 ;
    diff = rel_tol+1 ;
    while ((norm(diff) > rel_tol) && (n <
n_iterations))
        d = gradf(X) ;
        tmp = norm(d) ;
        if(tmp>1)
            d = d/tmp ;
        end
        h = @(l) f(X-l*d) ;
        alpha1 = 0 ; y1 = h(alpha1) ;
        alpha2 = step ; y2 = h(alpha2) ;
        while (y1 > y2)
            alpha1 = alpha2 ;
            y1 = y2 ;
            alpha2 = alpha1+step ;
            y2 = h(alpha2) ;
        end
        lopt = fminbnd(h, 0, alpha2) ;
        diff = lopt * d ;
        X = X - diff ;
        if (return_iterates)
            Xres = [Xres, X] ;
        end
    end
    if (~return_iterates)
        Xres = X ;
    end
end
```

```
clear
clc
clf
hold on

alpha = 2 ;
beta = 3 ;
omega = diag([alpha, beta]) ;

graph_f1 = @(x,y) alpha*x.^2 + beta*y.^2 ;
f1 = @(X) graph_f1(X(1),X(2)) ;
grad_f1 = @(X) 2*omega*X ;


graph_f2 = @(x,y) (alpha*x.^2 +
beta*y.^2).*(2+sin(x.^2+y.^2)) ;
f2 = @(X) graph_f2(X(1), X(2)) ;
grad_f2 = @(X) 2*omega*X * (2+sin(X'*X)) +
X'*omega*X *(2*X*cos(X'*X)) ;

zoom =  5 ;
[X,Y] = meshgrid(-zoom:0.1:zoom,
-zoom:0.1:zoom) ;
Z = graph_f2(X,Y) ;
mesh(X,Y,Z);

% optimisation
x0 = [3;4] ;
step = .1 ;
rel_tol = 1e-3 ;
n_iterations = 1000 ;
[Xres] = desc_grad_opt(f2, grad_f2, x0, step,
rel_tol, n_iterations, true) ;

% visu 3D
N = size(Xres,2) ;
ZZ = zeros(1,N) ;
for i=1:N
    ZZ(i) = f2(Xres(:,i)) ;
end

plot3(Xres(1,:), Xres(2,:), ZZ, 'r')
```