```
% Take Home Exam 2 - Exercise 4 - Anita Mezzetti

% goal: find the Heston parameters that minimize the root-mean-squared error
%       of the differences between the Heston prices and the observed prices
clear all
clc

global Price K T r S                % global variables

load('Call_20050103.mat')           % load data
% columns = call prices, strikes, time-mat (days) and implied volatilities

Price = Call_20050103(:,1);         % call price
K = Call_20050103(:,2);             % strikes
n_days = 252;                       % business days in a year
T = Call_20050103(:,3)/n_days;      % time to maturity
% ImpliedVol = Call_20050103(:,4);    % implied volatility (not needed)

r = 0.015;                          % risk free interest rate
S = 1202.10;                        % initial stock prices

% Initial parameters and limiters
% structure [theta, kappa, sigma, rho,  V]
par_start = [0.04, 1.50, 0.30, -0.60, 0.0441];
lower_bound = [-Inf, 0, 0, -1, 0];
upper_bound = [Inf, Inf, Inf, 1, Inf];

% Optimization routine:
[x, fval, exitflag, output] = fminsearchcon...
    (@distance_prices, par_start, lower_bound, upper_bound);

% Output:
disp('Optimal parameters:')
fprintf(['Nu = %f\n','Kappa = %f\n','Sigma = %f\n'...
    ,'Rho = %f\n', 'V = %f\n\n'],x(1), x(2), x(3), x(4), x(5));
fprintf(['Optimization routine iterations: %f'], output.iterations);
fprintf('\n\n Message:\n')
disp(output.message)


% Distance prices function:
function [error] = distance_prices(x)

% parameters:
nu = x(1);
```

```
kappa = x(2);
sigma = x(3);
rho = x(4);
V = x(5);

global Price K T r S    % global variables

error = 0;

for i = 1:length(Price)
    call_heston_price = Call_Heston(K(i),T(i),r,nu,kappa,sigma,rho,S,↙
V);
    error = error + (Price(i)-call_heston_price)^2;
end

% square eroor: do not add anything here
% error = sqrt(error) % root square error
error = sqrt(mean(error));  % root mean square error

end
```