fminunc Tutorial

AEROSP 584 - ANAV - Fall 2021

Author: Juan Paredes

1 Unconstrained optimization problem

Let $x \in \mathbb{R}$ and let $F\colon \mathbb{R} \to \mathbb{R}$ be the cost function such that

$$F(x) \triangleq (\mu(x - x_0)^2 + y_0)^2 \geq 0, \tag{1}$$

where $\mu, x_0, y_0 \in \mathbb{R}$. Find $x_{\text{min}} \in \mathbb{R}$ such that

$$x_{\text{min}} \in \arg\min_{x \in \mathbb{R}} F(x). \tag{2}$$

2 Solution setup using fminunc

First, let's set up a Matlab function that represents the cost function shown in Equation (1). The following function (costFun) is used for that purpose:

```
function F = costFun(x, x0, y0, mu)
    F = (mu.*((x - x0).^2) + y0);
    F = F.^2;
end
```

In costFun, $x_0 =$ x0, $y_0 =$ y0, and $\mu =$ mu. In your main function, suppose that x0, y0 and mu are defined. Then, add the following lines to the code:

```
1 F = @(x) costFun(x, x0, y0, mu);
2 xinit = -10;
3 options = optimoptions(@fminunc,'OptimalityTolerance',1e-12);
4 [xmin, fval] = fminunc(F, xinit,options);
```

In line 1, function F is being defined, such that F(x) = costFun(x, x0, y0, mu). Since x0, y0, and mu are parameters that won't change during execution, this simplifies the way in which code is presented.

In line 2, the variable $x_{\text{init}} =$ xinit is defined. This represents an initial guess on the value of x_{min} and will determine the solution you obtain in the presence of multiple local minima.

In line 3, some fminunc options are changed. In this case, 'OptimalityTolerance' is reduced to 10^{-12} to increase the accuracy of the results. However, decreasing it too much may slow down the code, so one should be careful when choosing this value. Other options are available and can be reviwed in the fminunc documentation.

Finally, line 4 shows how to use fminunc to obtain a minimizer of function F given the initial guess x_{init} and the options set in line 3. xmin represents x_{min} and fval = $F(x_{\text{min}})$.

3 Numerical results

For all numerical simulations, $\mu = 0.5$. We'll study 3 different cases. The code used for cases 1 and 2 is displayed in Appendix A and the code used for case 3 is displayed in Appendix B. Note that in the code of Appendix A, OutputFcn is added to the fminunc options. This specifies a function that will be run at each iteration of the optimization function. In this case, it'll be used to store the minimizer of the algorithm at each iteration in the global variable h.

Case 1: $x_0 = 3, y = 0$

Figure 1 shows the result from running fminunc using $x_{\text{init}} = -10$. Note that, with a higher value of 'Optimality-Tolerance', the value of $x_{\min}$ could be farther from 3 (the minimizer) due to how "flat" the cost function is around the minimizer. This is also illustrated in Figure 2, where the logarithmic error between the minimizer at each iteration and the true minimizer x_0 diminisher in a linear fashion, unlike the minimization performed in the next case (case 2).

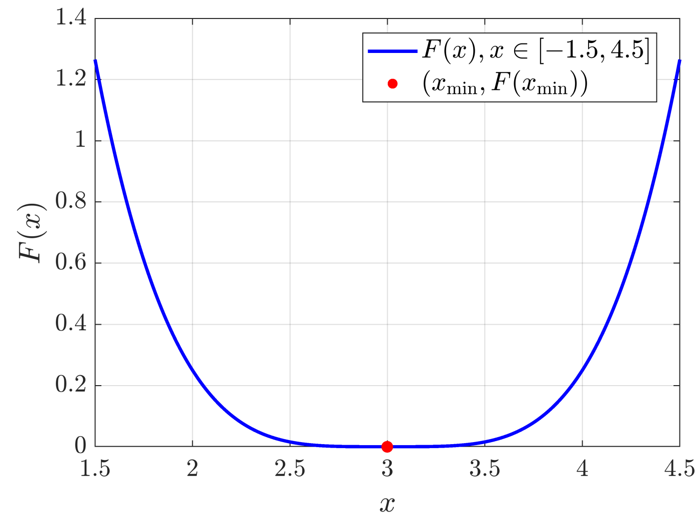

Figure 1: Optimization results for Case 1 using $x_{\text{init}} = -10$.

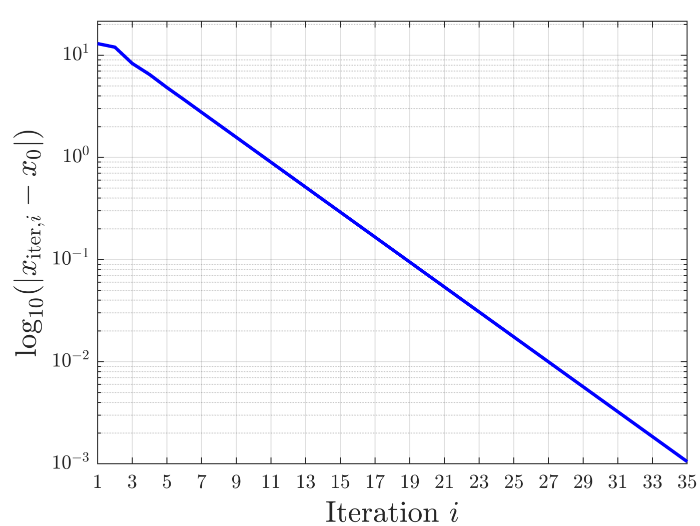

Figure 2: fminunc logarithmic error vs iteration number for Case 1 with $x_{\text{init}} = -10$. $x_{\text{iter},i}$ is the fminunc minimizer at iteration i.

Case 2: $x_0 = 3, y = 2$

Figure 3 shows the result from running fminunc using $x_{\text{init}} = -10$. Since the cost function is more concave that the one in Case 1 around the minimizer, an accurate solution can be reached even with a higher value of 'OptimalityTolerance'. Figure 4 shows a faster reduction of the logarithmic error between the minimizer at each iteration and the true minimizer than the one observed in Case 1 (even the number of iterations required is lower in the second case.)

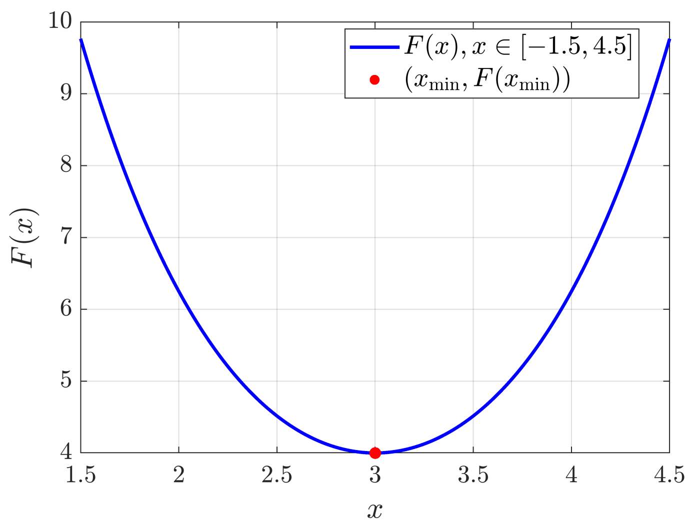

Figure 3: Optimization results for Case 2 using $x_{\text{init}} = -10$.

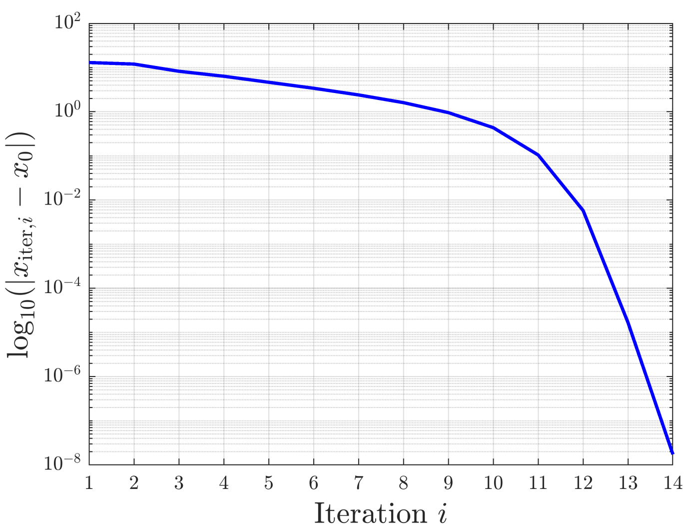

Figure 4: fminunc logarithmic error vs iteration number for Case 2 with $x_{\text{init}} = -10$. $x_{\text{iter},i}$ is the fminunc minimizer at iteration i.

Case 3: $x_0 = 3, y = -2$

Figure 5 shows the result from running fminunc using $x_{\text{init}} \in \{-10, 10\}$. Since the cost functions has two minimizers at 1 and 5, the value chosen for x_{init} affects the solution that fminunc reaches. To illustrate this, Figure 6 shows the minimizer obtained for a grid of values of x_{init}, such that $x_{\text{init}} \in \{-7, -5, \ldots, 11, 13\}$. fminunc tends to move in the direction of the minimizer that is "closer" to the initial guess. However, one should be careful on how "closer" is interpreted since this will depend on the cost function.

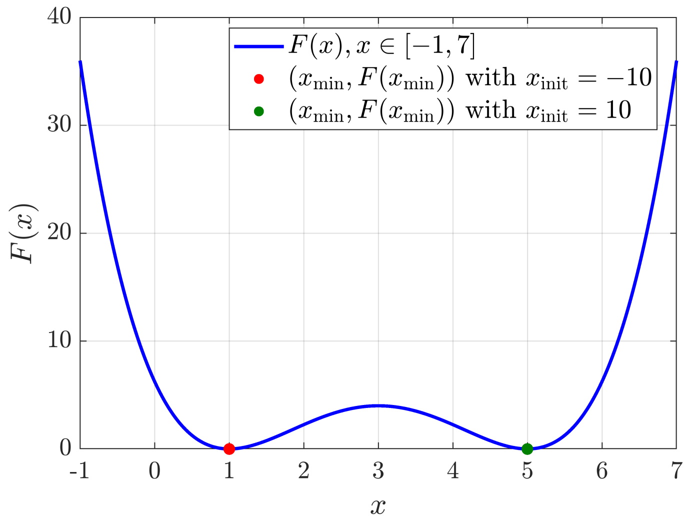

Figure 5: Optimization results for Case 3 using $x_{\text{init}} \in \{-10, 10\}$.

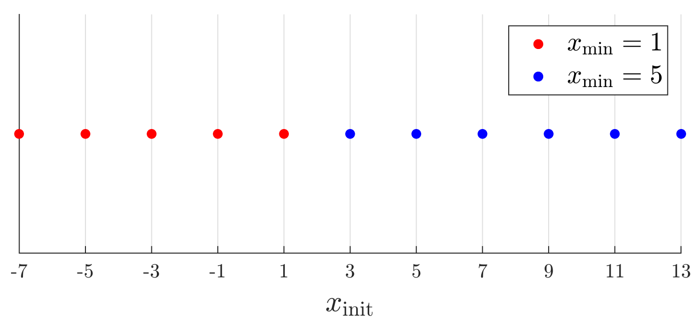

Figure 6: Grid of initial points and minimizer results for Case 3, such that $x_{\text{init}} \in \{-7, -5, \ldots, 11, 13\}$. Red initial points yield $x_{\min} = 1$ and blue initial points yield $x_{\min} = 5$.

Appendix A fminunc_C1_C2.m

```
close all
clear all
clc

%% Optimization setup
x0 = 3;
y0 = 0;
%y0 = 2;
mu = 0.5;

global h;
h.x = [];
h.fval = [];

F = @(x) costFun(x, x0, y0, mu);
options = optimoptions(@fminunc,'OptimalityTolerance',1e-12,'OutputFcn',@outfun);
xinit = -10;

%% Optimization function
[xmin, fval] = fminunc(F, xinit,options);

%% Plotting solutions
xr =1.5:0.01:4.5;

figure(1)
plot(xr, F(xr), 'b', 'linewidth', 2)
hold on
scatter(xmin,F(xmin), 50,'r','fill')
hold off
xlim([1.5 4.5])
set(gca,'FontSize',15)
set(gca,'TickLabelInterpreter','latex');
xlabel('$x$','fontsize',18,'interpreter','latex')
ylabel('$F(x)$','fontsize',18,'interpreter','latex')
legend({'$F(x), x \in [-1.5, 4.5]$','$(x_{\rm min}, F(x_{\rm min}))$'},'fontsize',16,'interpreter','latex')
grid on

figure(2)
iter = 1:length(h.x);
semilogy(iter, abs(h.x-x0),'b','linewidth',2)
xlim([min(iter) max(iter)])
xticks(1:2:max(iter))
%xticks(1:max(iter))
set(gca,'FontSize',12)
set(gca,'TickLabelInterpreter','latex');
xlabel('Iteration $i$','fontsize',18,'interpreter','latex')
ylabel('$\log_{10} (|x_{{\rm iter},i} - x_0|)$','fontsize',18,'interpreter','latex')
grid on

%%
function stop = outfun(x,optimValues,state)
    global h
```

```
        stop = false;
        switch state
            case 'iter'
                h.fval = [h.fval; optimValues.fval];
                h.x = [h.x; x];
            otherwise
        end
    end
```

Appendix B fminunc_C3.m

```
close all
clear all
clc

%% Optimization setup
x0 = 3;
y0 = -2;
mu = 0.5;

F = @(x) costFun(x, x0, y0, mu);
options = optimoptions(@fminunc,'OptimalityTolerance',1e-12);
xinit1 = -10;
xinit2 = 10;

%% Optimization function
[xmin1, fval1] = fminunc(F, xinit1,options);
[xmin2, fval2] = fminunc(F, xinit2,options);

%% Optimization over range
xinit = -7:2:13;
xmin = zeros(1,length(xinit));

for ii = 1:length(xinit)
    xmin(ii) = fminunc(F, xinit(ii),options);
end

xinit_C1 = abs(xmin-xmin1);
xinit_C2 = abs(xmin-xmin2);

xR = xinit(xinit_C1 <= xinit_C2);
xB = xinit(xinit_C1 > xinit_C2);

%% Plotting solutions
xr = -1:0.01:7;

figure(1)
plot(xr, F(xr), 'b', 'linewidth', 2)
hold on
scatter(xmin1,F(xmin1), 50,'r','fill')
scatter(xmin2,F(xmin2), 50,[0, 0.5, 0],'fill')
hold off
xlim([-1 7])
set(gca,'FontSize',15)
set(gca,'TickLabelInterpreter','latex');
xlabel('$x$','fontsize',18,'interpreter','latex')
```

```
ylabel('$F(x)$','fontsize',18,'interpreter','latex')
legend({'$F(x), x \in [-1, 7]$','$(x_{\rm min}, F(x_{\rm min}))$ with $x_{\rm init} = -10$', '$(x_{\rm min}, F(x_{\rm min}))$ with $x_{\rm init} = 10$'},'fontsize',16,'interpreter','latex')
grid on

figure(2)

scatter(xR, zeros(1,length(xR)),35,'r','fill')
hold on
scatter(xB, zeros(1,length(xB)),35,'b','fill')
hold off
xlim([min(xinit) max(xinit)])
ylim([-0.5 0.5])
set(gca,'FontSize',12)
set(gca,'TickLabelInterpreter','latex');
xlabel('$x_{\rm init}$','fontsize',18,'interpreter','latex')
yticks([])
xticks(xinit)
legend({'$x_{\rm min} = 1$','$x_{\rm min} = 5$'},'fontsize',16,'interpreter','latex')
grid on
```