1. Error in Finite Difference Approximations

a.

b.

## Define error term function for part a

function compute\_err\_a(x, h)

return h

end

## Define ranges for x and h

x\_range = 1:10

h\_range = 0.1:0.01:1.0

## Compute error at each combination of x and h\_range

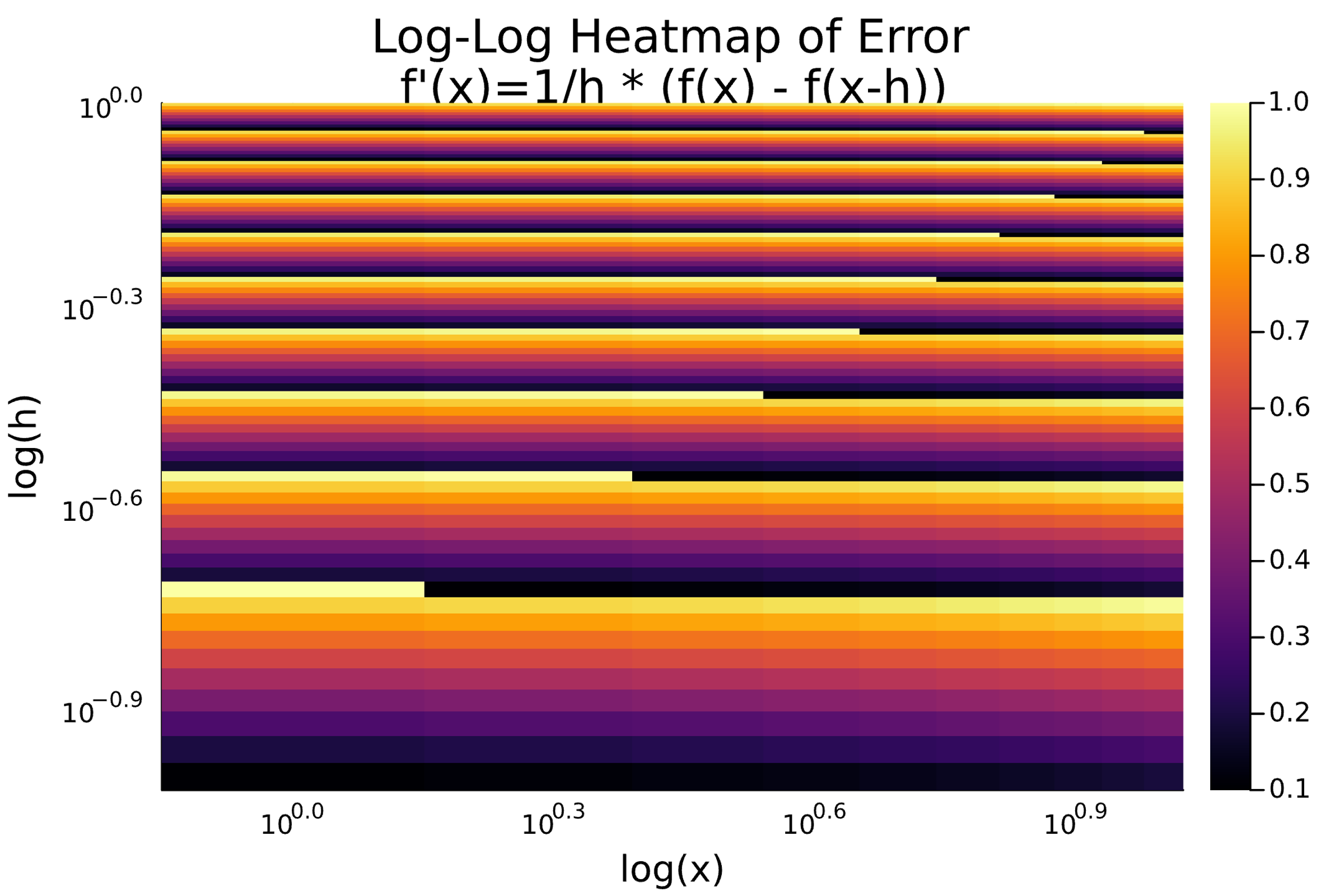
err\_a = [compute\_err\_a(x, h) for x in x\_range, h in h\_range]

## Plot log-log heatmap

heatmap(x\_range, h\_range, err\_a, xlabel="log(x)", ylabel="log(h)",

title="Log-Log Heatmap of Error\nf'(x)=1/h \* (f(x) - f(x-h))",

xscale=:log10,yscale=:log10)



c)

**Plotting the heatmap**

## Define error term function for part c

function compute\_err\_c(x, h)

return 0

end

## Define ranges for x and h

x\_range = 1:10

h\_range = 0.1:0.01:1.0

## Compute error at each combination of x and h\_range

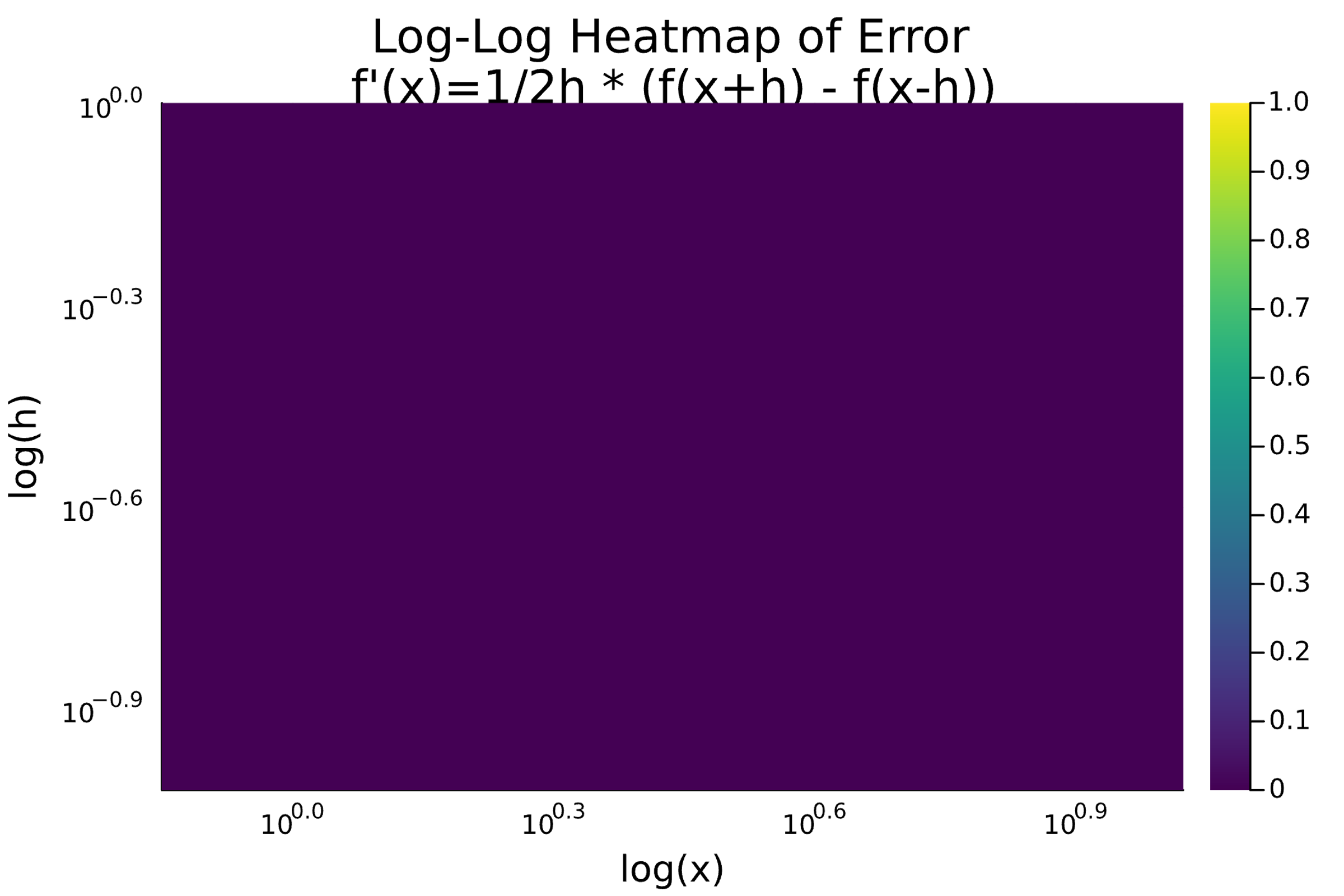
err\_c = [compute\_err\_c(x, h) for x in x\_range, h in h\_range]

## Plot log-log heatmap

heatmap(x\_range, h\_range, err\_c, xlabel="log(x)", ylabel="log(h)",

title="Log-Log Heatmap of Error\nf'(x)=1/2h \* (f(x+h) - f(x-h))",

xscale=:log10,yscale=:log10, color=:viridis)



2. Using Convex.jl and SCS for Non-negative Least Squares

For this problem, I decided to use the “Wine Quality” dataset from UC Irvine’s Machine Learning repository to predict the quality rating of the red variant of the Portuguese “Vinho Verde” wine – citation below.

Cortez, Paulo, Cerdeira, A., Almeida, F., Matos, T., and Reis, J. (2009). Wine Quality. UCI Machine Learning Repository. https://doi.org/10.24432/C56S3T.

Using the following code, I tested the capability for prediction of the NNLS and the ULS on this dataset.

3. Illustration of Function Extrema

a. Define and illustrate a global function maximizer

A global maximizer of a function f(x) is a point x\* in the domain of f such that for all x in the domain of f. In other words, x\* is the point where the function attains its highest value in its entire domain. For example, let’s consider the function . We can tell that this function reaches its maximum at x = 0, when f(x) = 4. To illustrate this, let’s look at this graph defined in this Julia code:

using Plots

# Define the function

f(x) = -x^2 + 4

# Generate x values

x = -3:0.1:3

y = f.(x)

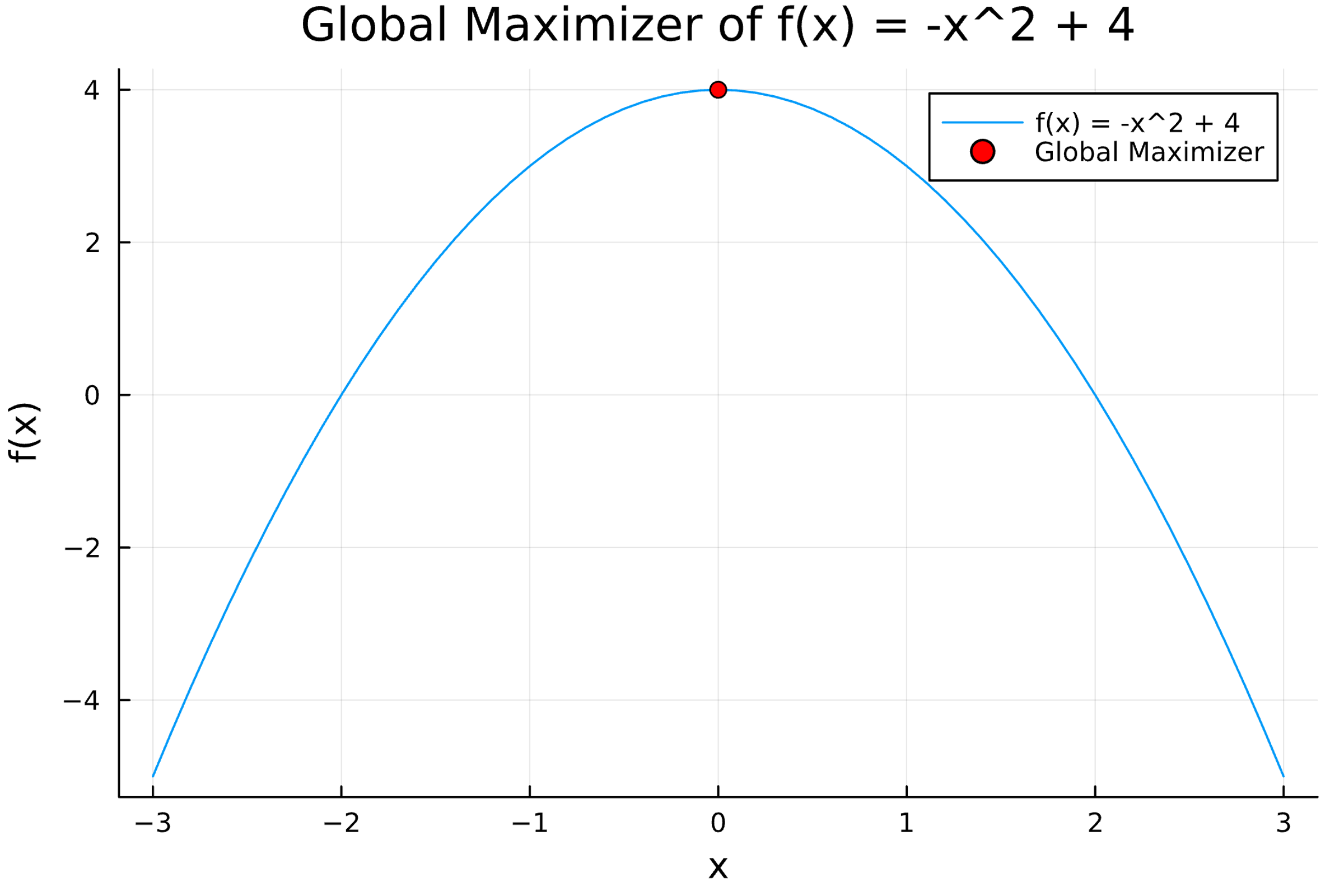
# Plot the function

plot(x, y, label="f(x) = -x^2 + 4", xlabel="x", ylabel="f(x)", title="Global Maximizer of f(x) = -x^2 + 4")

scatter!([0], [f(0)], label="Global Maximizer", color=:red, markersize=4)

# Show the plot

display(plot!())



The red dot represents the global maximum of the function, as it is the point where the function reaches its peak value of 4.

b. Define and illustrate a local function minimizer

A local minimizer of a function f(x) is a point x\* in the domain of f such that there exists a neighborhood around x\* where for all x in that neighborhood. This is distinct from the global minimizer, because while the global minimum is a local minimum, a local minimum may not be the global minimum.

Taking the function and using Julia to visualize it, we can observe the following:

# Define the function

g(x) = x^4 - 12\*x^2 + 2\*x

# # Generate x and y values for global function

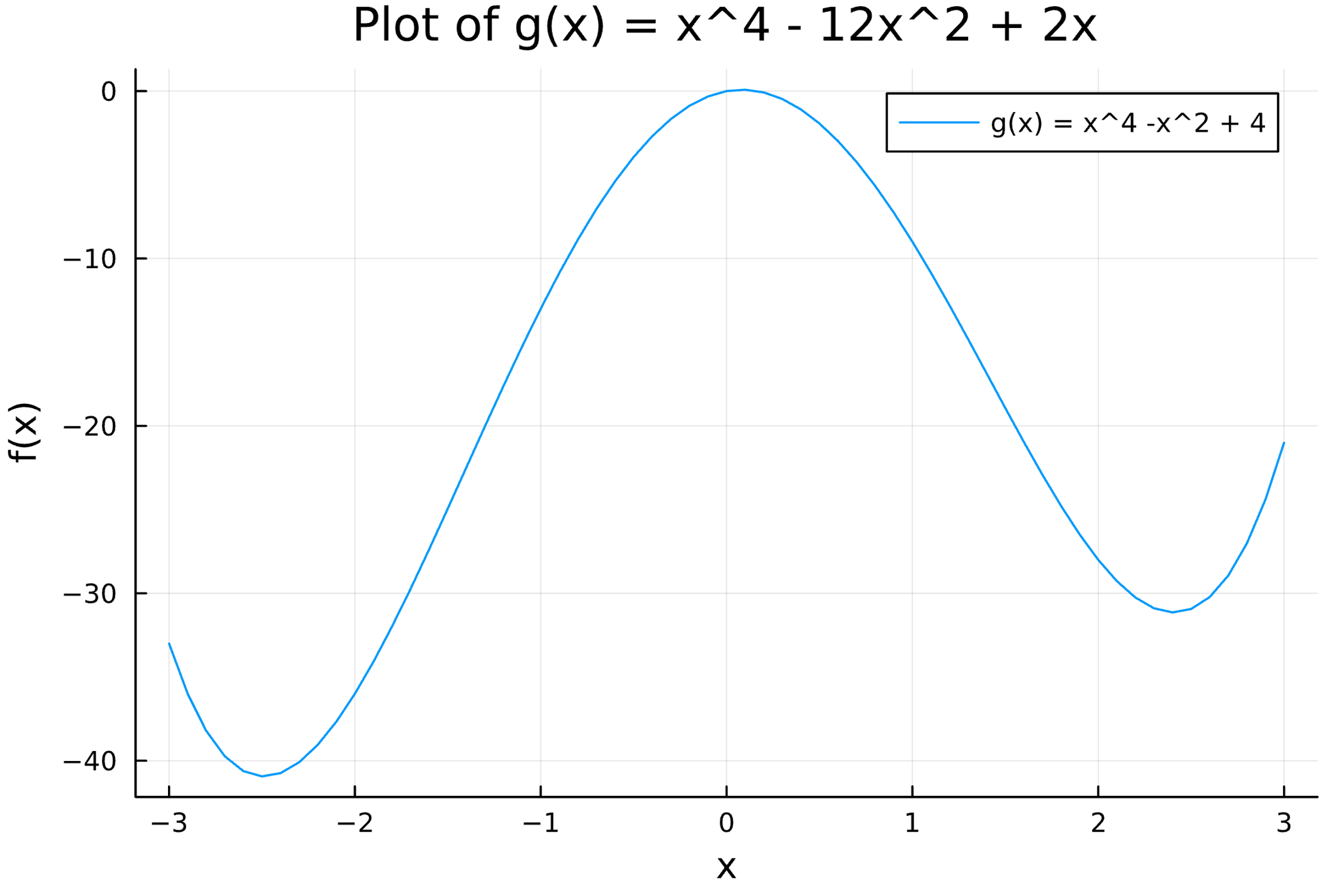
x = -3:0.1:3

y = g.(x)

# Show the plot

plot(x, y, label="g(x) = x^4 -x^2 + 4", xlabel="x", ylabel="f(x)", title="Plot of g(x) = x^4 - 12x^2 + 2x")

display(plot!())



This is a great plot to identify the local minimizers of this function as there are two troughs in the plot. If we take the neighborhoods as being inclusive, we can then define the local minimizers as the smallest value in each domain. I’ve implemented that using these lines in the Julia code above

# define neighborhoods

x1 = -3:0.1:0

y1 = g.(x1)

x2 = 0:0.1:3

y2 = g.(x2)

# Identify local minima

loc1 = x1[findmin(y1)[2]]

loc2 = x2[findmin(y2)[2]]

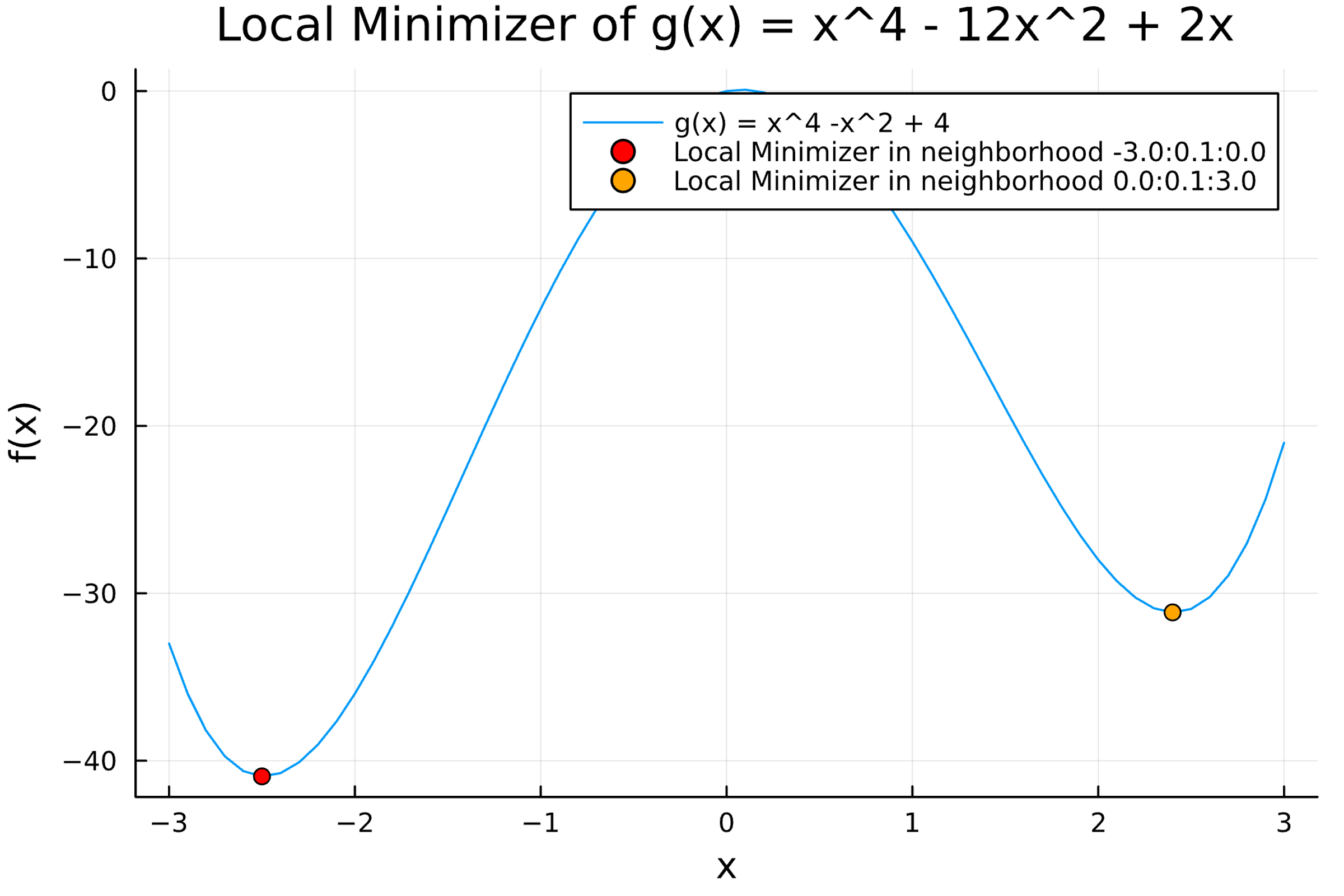
# Show the plot

plot(x, y, label="g(x) = x^4 -x^2 + 4", xlabel="x", ylabel="f(x)", title="Plot of g(x) = x^4 - 12x^2 + 2x")

scatter!([loc1], [g(loc1)], label="Local Minimizer in neighborhood $x1", color=:red, markersize=4)

scatter!([loc2], [g(loc2)], label="Local Minimizer in neighborhood $x2", color=:orange, markersize=4)

display(plot!())



Now when we look at this modified plot, there are two dots to represent the minimizers of the neighborhoods bounded by -3 <= x1 <= 0 and 0 <= x2 <= 3, which we have identified as being at the bottom of the troughs split across the graph. This also illustrates the point of the local minimizers not being the global minimizers, as the orange dot, the minimizer for the non-negative neighborhood, is larger than the minimizer for the negative neighborhood.