Verkfræðilegar Bestunaraðferðir

Day 6 - Group 2

T-423-ENOP

Arnar Gylfi Haraldsson
arnarh23@ru.is

Hafþór Árni Hermannsson
hafthorh20@ru.is

Ragnheiður Gná Gústafsdóttir
ragnheidurg@ru.is

May 7, 2024

Exercise 1

Description

Use the `fminunc` routine to minimize a multi-dimensional generalization of the Rosenbrock function of the form:

$$f(x) = \sum_{i=1}^{n} (1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2,$$

where $x = [x_1, x_2, \ldots, x_n]^T$. Consider $n = 3, 5$, and 10. Use the zero vector as a starting point in each case. Perform the same task using the `fminsearch` routine. Compare the number of function evaluations necessary to obtain a function value smaller than 10^{-10}.

Solution

To solve this, the inputs are simply plugged into the `fminunc` and `fminsearch` functions. For the `fminsearch` function, the maximum number of iterations and function evaluations had to be increased from their default values, as it didn't find a solution within those default values.
Below is a table showing the number of function evaluations needed to obtain a function value smaller than 10^{-10} for both algorithms and each value of n.

n	`fminunc`	`fminsearch`
3	140	301
5	258	930
10	748	4875

Table 1: Number of function evaluations required to reach f(x) < 10^{-10}

Exercise 2

Description

Use the `fmincon` routine to minimize the function

$$f(x) = \exp\left(x_1 + \frac{x_2}{2} + \cdots + \frac{x_n}{n}\right),$$

so that $\|x\| \leq 1$. Use the zero vector as a starting point. Perform the task for $n = 2, 3, 4$, and 20.

Solution

Once again the appropriate variables are simply input into the function `fmincon`. The constraint `norm(x)`-1 $\leq$ 0 had to be modified to `norm(x)`^{2}-1 $\leq$ 0. So that it's differentiable at 0.
For n = 2 the result is obtained with 22 function evaluations as $\begin{bmatrix} -0.8944 \\ -0.4472 \end{bmatrix}$, for n = 3 we get with 37 function evaluations that $X = \begin{bmatrix} -0.8571 \\ -0.4286 \\ -0.2857 \end{bmatrix}$, for n = 4, and 46 evaluations, argmin(f(x)) = $\begin{bmatrix} -0.8381 \\ -0.4191 \\ -0.2794 \\ -0.2095 \end{bmatrix}$ and finally for n = 20 we get with 190 function evaluations that argmin(f(x)) = $\begin{bmatrix} -0.7915 \\ -0.3958 \\ -0.2638 \\ -0.1979 \\ -0.1583 \\ -0.1319 \\ -0.1131 \\ -0.0989 \\ -0.0879 \\ -0.0792 \\ -0.0720 \\ -0.0660 \\ -0.0609 \end{bmatrix}$

Table 2: The parameters of the final iteration for each n

n	Iter	F-count	f(x)	Feasibility	First-order Optimality	Norm of Step
2	6	22	3.269219e-01	0.000e+00	1.000e-09	2.718e-07
3	8	37	3.114032e-01	0.000e+00	2.628e-09	5.456e-08
4	8	46	3.032639e-01	0.000e+00	5.784e-09	5.483e-08
20	8	190	2.826931e-01	0.000e+00	6.938e-09	5.692e-08

Exercise 3

Description

Use the `lsqnonlin` routine from Matlab Optimization Toolbox to find the parameters a, b, c, and d of the function

$$g(t) = a \cdot \exp(-b \cdot t) + c \cdot \exp(-d \cdot t),$$

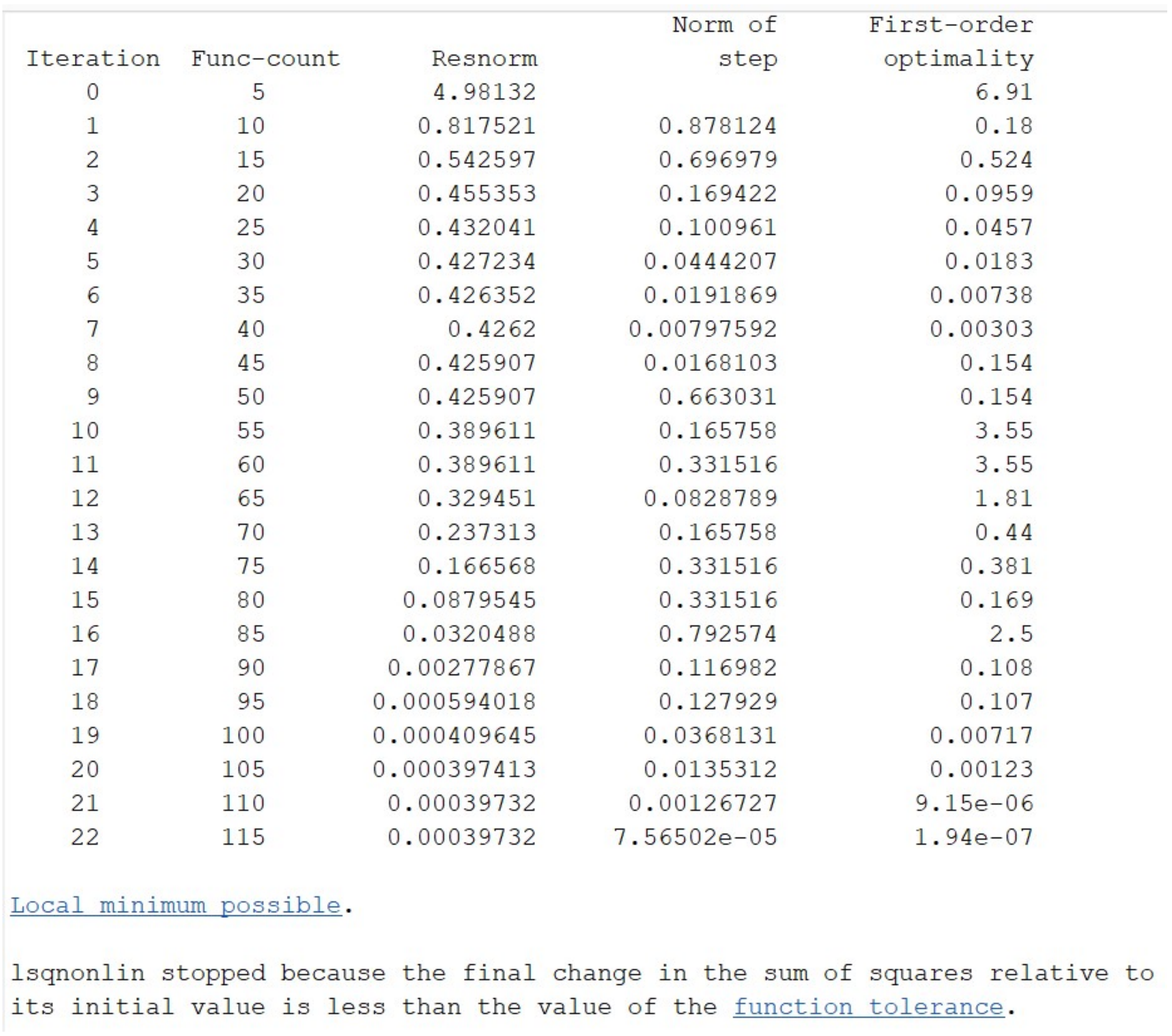
that approximates the following set of data: {(1,2.3743), (2,1.1497), (3,0.7317), (4,0.5556), (5,0.4675), (6,0.4157), (7,0.3807), (8,0.3546), (9,0.3337), (10,0.3164)} as well as possible in the least-square sense. Lower and upper bounds for parameters are 0 and 10. The initial parameter values are $a = b = c = d = 1$. Plot the data, and the approximation function for initial and optimized parameters.

Functionality

The `lsqnonlin` function is a nonlinear least-squares solver, it solves nonlinear least-squares curve fitting problems that may be subject to constraints. Firstly we defined all the parameters, the function, data, lower and upper bounds and the initial parameters. The data was split into the time, t, and the values, y. Then the values were subtracted from the initial function to get the residual that was then fed into the `lsqnonlin` function alongside the initial data points, the bounds and a few options such as the tolerances for X and Fun which were set to 1e-8. Then it was plotted up.

Result

When the code was run the output is shown in figure 1, as can be seen there it took the `lsqnonlin` function 22 iterations and 115 functions to find a function where the sum of squares satisfied the function tolerance, set at 1.0e-08. Figure 2 shows the result plotted up with the inital para,eters and the data to which they were to be fitted.



```
                                          Norm of      First-order
 Iteration  Func-count     Resnorm          step        optimality
     0           5         4.98132                            6.91
     1          10        0.817521        0.878124            0.18
     2          15        0.542597        0.696979           0.524
     3          20        0.455353        0.169422          0.0959
     4          25        0.432041        0.100961          0.0457
     5          30        0.427234       0.0444207          0.0183
     6          35        0.426352       0.0191869         0.00738
     7          40          0.4262      0.00797592         0.00303
     8          45        0.425907       0.0168103           0.154
     9          50        0.425907        0.663031           0.154
    10          55        0.389611        0.165758            3.55
    11          60        0.389611        0.331516            3.55
    12          65        0.329451       0.0828789            1.81
    13          70        0.237313        0.165758            0.44
    14          75        0.166568        0.331516           0.381
    15          80       0.0879545        0.331516           0.169
    16          85       0.0320488        0.792574             2.5
    17          90      0.00277867        0.116982           0.108
    18          95     0.000594018        0.127929           0.107
    19         100     0.000409645       0.0368131         0.00717
    20         105     0.000397413       0.0135312         0.00123
    21         110      0.00039732      0.00126727        9.15e-06
    22         115      0.00039732     7.56502e-05        1.94e-07

Local minimum possible.

lsqnonlin stopped because the final change in the sum of squares relative to
its initial value is less than the value of the function tolerance.
```

Figure 1: The output of the `lsqnonlin` function.

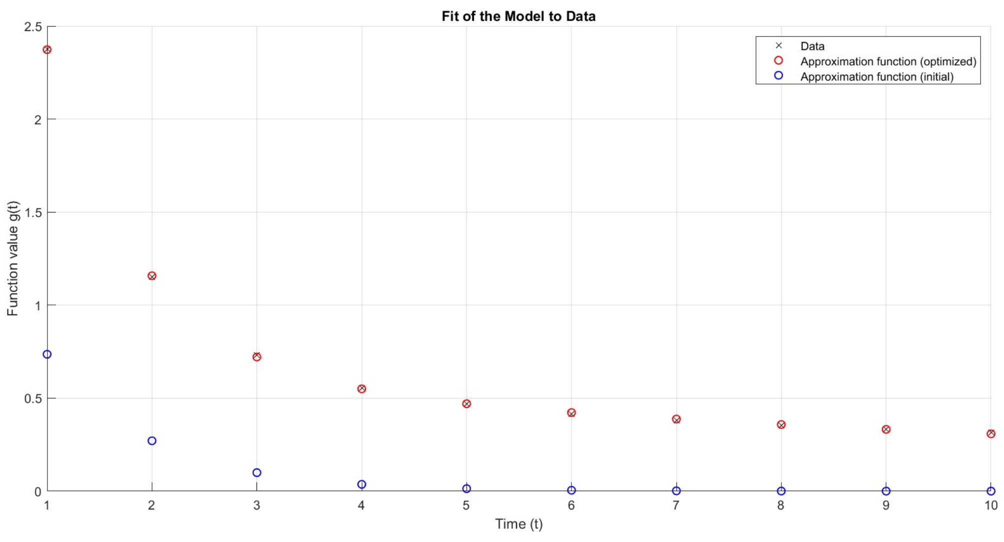


Figure 2: The result plotted up.

Exercise 4

Description

Consider a function `black_box` that takes a vector argument $x = [x_1, \ldots, x_n]^T$ and returns a vector $f(x) = [f_1(x), \ldots, f_n(x)]^T$. Use Matlab Optimization Toolbox to minimize the following expression $\max\{f_1(x), \ldots, f_n(x)\}$ under the following constraints: $x_1 + \ldots + x_n = 1$, and $x_j \geq 0$ for $j = 1, \ldots, n$. The starting point is $x_0 = [1/n, \ldots, 1/n]^T$. Plot initial and optimized function values. Consider the following cases $n = 3, 5, 10$, and 20.

Functionality

The implementation utilizes Matlab's `fmincon` function from the Optimization Toolbox to minimize the maximum value of a function $f(x)$ under constraints. This is applied to vectors of different sizes, specifically $n = 3, 5, 10$, and 20. The constraints ensure that the elements of x sum to 1 and each element is non-negative, starting from a uniform initial guess.

- **Optimization setup:** The script uses the 'sqp' algorithm and displays iterations.
- **Constraints:** Linear equality constraints are defined to ensure the elements of x sum to 1.
- **Bounds:** Non-negative bounds are applied to each element of x.
- **Objective function:** The function $f(x)$ is maximized by minimizing its maximum element.

Result

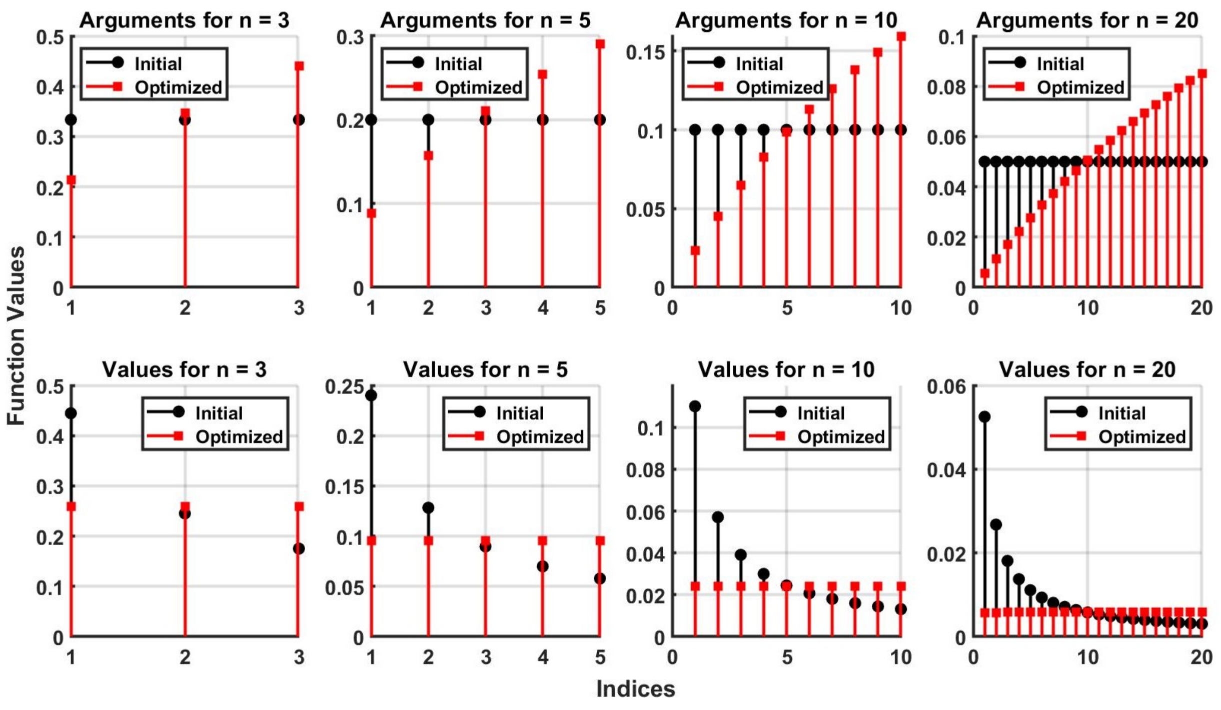


Figure 3: Exercise 4 results