

Computational Optimization  
Laboratory 7 - 3/25/08 - Nonlinear Constraints  
To be completed in class and hand in by 3/28.

We are going to continue to look at the routine for constrained minimization called “fmincon”. Today we will be looking at the case of minimizing nonlinear functions with only nonlinear constraints. Be sure to answer all of the questions asked; note there are many small questions asked within this lab for which an answer should be supplied in your write up. You don’t have to turn in a diary or your m files. Just supply the solutions found for each run and your answer to the questions.

1. Consider the problem:

$$\begin{aligned} \min_{x_1, x_2} \quad & e^{x_1}(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1) \\ \text{subject to} \quad & x_1x_2 - x_1 - x_2 \leq -1.5 \\ & x_1x_2 \geq -10 \end{aligned}$$

The optimizer expects the problem to be a minimization problem subject to constraints of the form  $G(x) \leq 0$ . Thus, we rewrite the problem:

$$\begin{aligned} \min_{x_1, x_2} \quad & e^{x_1}(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1) \\ \text{subject to} \quad & x_1x_2 - x_1 - x_2 + 1.5 \leq 0 \\ & -x_1x_2 - 10 \leq 0 \end{aligned}$$

The provided m-file (f7.m) returns the objective function and optionally gradient. The “nargout” option (see below) is used such that the gradient will only be calculated when needed. Note that a temporary parameter is used to avoid repeated calculations in this file.

2. We now need to create an m-file for the nonlinear constraints (f7con.m). Be sure to note the form of the matrix of gradients of the constraints.

$$\begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_2}{\partial x_1} \\ \frac{\partial g_1}{\partial x_2} & \frac{\partial g_2}{\partial x_2} \end{bmatrix}$$

The m-file has the following form.

```
function [c,ceq,gc,gceq]=f7con(x)
c=[...];
ceq=[];
if nargout>2
    gc=[...];
    gceq=[];
end
```

Since the problem has no equality constraints, the equality constraints ceq and gceq are defined to have null values. The variable nargout contains the number of arguments given to the function. The gradients are only calculated if the number of arguments is greater than 2.

- Verify that the constraint function, `f7con.m` available from the course web page is correct. Compute the Jacobian by hand. Compute the objective, gradient and Jacobian values of `f7` at  $x_0 = [-1, 1]$ . Verify by hand that your `f7.m` and `f7con.m` work appropriately at the point  $x_0$ . Make sure you understand the effect of supplying multiple output arguments to `f7` and `f7con`. For example, what is the difference between the two calls to `f7` given below.

```
x0=[-1;1];
[func]=f7(x0);
```

```
[func,grad]=f7(x0)
```

Compute the values and gradient of the constraints at  $x_0$  by hand. Verify that `f7con` returns the same value by executing the following command:

```
[fc,fceq,gc,gceq]=f7con(x0);
```

`fc` and `gc` contain the value and Jacobian (gradient matrix) of the nonlinear equality constraints. If `gc` is active the value should be 0 within numeric tolerances.

- Make two sets of options, one to utilize user supplied gradients and one which uses finite differences. Note that for the user supplied gradients, you must turn on two options!

```
opts1=optimset('display','iter','largescale','off','diagnostics','on');
opts2=optimset(opts1,'gradobj','on','gradconstr','on','derivativecheck','on');
```

- Since there are no linear constraints or bounds, you should set the parameters as:

```
A=[]
b=[]
Aeq=[]
beq=[]
lb=[]
ub=[]
```

- Invoke the optimization routine:

```
x0 = [-1;1];
[x,fval,exitflag,output,lambda,grad,hessian]=fmincon('f7',x0,A,b,
    Aeq,beq,lb,ub,'f7con',opts2)
```

You should get that the optimal objective value is 0.02364. What is the solution for  $x$  found?

- Write down the KKT optimality conditions in Lagrangian multiplier form.

Primal Feasibility:

Dual Feasibility:

Complementarity:

8. You can use matlab to check the optimality conditions for you. To do this you will need to check the function, gradient, Jacobian, and Lagrangian multipliers. You can calculate them as follows:

```
[fun, grad]=f7(x)
[fc,fcged,gc,gced] = f7con(x)
```

In this case "fun" and "grad" are the objective value and gradient computed at  $x$ .  $fc$  and  $gc$  are the value and Jacobian of the inequality constraint vector evaluated at  $x$ . In this case, "fcged" and "gced", would be the constraint vector and Jacobian for the equality constraints. In this problem there are none, so they are empty sets.

Type lambda. You will see the lambda is a structure containing the Lagrangian multipliers for each possible type of constraint. In this example, the only constraints are nonlinear inequality constraints, so the only defined components of lambda would be lambda.ineqnonlin. To see them type lambda.ineqnonlin.

9. Now we can use these values to check the KKT conditions. For primal feasibility,  $fc$  is the value of  $g(x)$ . Does the answer agree with matlabs report that two constraints are active? How does knowing that the computer uses floating point arithmetic with limited precision affect your answer?

For dual feasibility, lambda.ineqnonlin should be nonnegative and the gradient of the Lagrangian with respect to  $x$  should be 0. We can evaluate these quantity numerically:

```
grad+gc*lambda.ineqnonlin
```

Note that the constraint term is added instead of subtracted since the matlab convention is  $g(x) \leq 0$ . Is the solution dual feasible? How does knowing that the computer uses floating point arithmetic with limited precision affect your answer?

For complementarity we would like the dot product of the multipliers and the constraints to be 0. To check this numerically,

```
fc'*lambda.ineqnonlin
```

Does complementarity hold? How does knowing that the computer uses floating point arithmetic with limited precision affect your answer?

10. To check FONC, SONC, and SOSC, we need to know the Jacobian of the active constraints. In this case  $gc$ , is the Jacobian of the active constraints since all constraints are active.  $GC$  has full rank, so we know the gradient of the constraints are independent thus the solution is a regular point (LICQ holds), so FONC hold. Also we know that the only point in the null space is 0. So the SONC and SOSC would hold vacuously. Thus there is no need to actually compute the second derivative of the Lagrangian.

11. Lets add bounds constraints and try again. Say we wanted to solve the following problem:

$$\begin{aligned} \min_{x_1, x_2} \quad & e^{x_1}(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1) \\ \text{subject to} \quad & x_1x_2 - x_1 - x_2 + 1.5 \leq 0 \\ & -x_1x_2 - 10 \leq 0 \\ & 0 \leq x_1 \\ & 0 \leq x_2 \end{aligned}$$

We can just define upper and lower bounds vectors. Let's solve using analytical gradients.

```
lb=[0;0];
ub=[ ]; % Empty because the bound is infinity.
x0 = [-1;1]
[x,fval,exitflag,output,lambda,grad,hessian]=fmincon('f7',x0,A,b,
    Aeq,beq,lb,ub,'f7con',opts2)
```

What is the optimal solution and which constraints are active? Did adding the bounds make the problem easier or harder?

12. What are the KKT conditions for the problem defined in 11?

Matlab returns the Lagrangian multipliers. To find the ones for the types of constraints in this problem, type:

```
lambda.ineqnonlin
lambda.lower
```

13. Write down the KKT conditions in Lagrangian multiplier form.

Write down the matlab commands need to check primal feasibility.

Write down the matlab commands needed to check dual feasibility.

Write down the matlab commands needed to check complementarity.

You can check these as above but now.

Is the point a KKT point?

14. To examine the FONC, SONC, and SOSC, we need to know the Jacobian for the active constraints. Show that these are

`[gc(1,:); -1 0]`

Once again there are two active constraints in a two dimensional space. So the only point in the null space is the zero vector. So we know we have a regular point and we know that the SONC and SOSC hold vacuously.

15. **For practice only. Do not turn in.** Solve the following problem in Matlab using analytical gradients. Be sure to try several different starting points. Sketch the problem and label each local and global minimizer. Check if SONC and SOSC hold at the solutions you find.

$$\begin{array}{ll} \min_{x_1, x_2} & x_1 + x_2 \\ \text{subject to} & (x_1 - 1)^2 + x_2^2 \leq 2 \\ & (x_1 + 1)^2 + x_2^2 \geq 2 \end{array}$$