

### Matlab Assignment 3, due Friday, April 6. Using Matlab to solve max/min problems

Matlab has some sophisticated max/min tools in its optimization toolbox which we will not describe here. Instead we will show how Matlab can be used to solve max/min problems following the techniques we have learned in class. Normally Matlab treats variable names such as  $x$  and  $y$  as constants, either numbers or lists of numbers or arrays of numbers. But it is possible to make Matlab think of  $x$  and  $y$  as true variables and manipulate expressions symbolically, for example differentiating and integrating. You specify a variable using the `syms` statement. For example, the following statements set up the function  $f(x, y) = x \cos(x + y) - e^{xy}$  and calculate  $\partial f / \partial x$  and the antiderivative with respect to  $y$ .

```
>> syms x y
>> f = x*cos(x+y)-exp(x*y)
>> fx = diff(f,x)
>> int(f,y)
```

Evaluation of functions is more cumbersome. You need to use the `subs` command. Calculate  $f(0, 1)$  by

```
>> subs(f, {x y}, {0 1})
```

which gives a symbolic answer or to get an actual number use the command `double` which converts a symbolic number to a number

```
>> double(subs(f, {x y}, {0 1}) )
```

Alternatively, you could make  $f$  into an inline function which can be evaluated more easily.

```
>> fin = inline(vectorize(f), 'x','y')
>> fin(0,1)
```

You can have Matlab solve a system of equations using the `solve` command. Matlab will try to solve symbolically, much as you have learned to do when solving a system of equations arising from finding critical points or using Lagrange multipliers. If it succeeds, it will find a list of all solutions. But the equations may be too complicated to solve symbolically, in which case it will try to find numerical solutions. In this case it may not find all solutions, so you should be wary. For example

```
>> syms x y
>> f = x - 3/y
>> g = 3*y+x-7
>> [a b] = solve(f,g)
```

will solve the equations  $f(x, y) = 0$ ,  $g(x, y) = 0$  exactly and find the two symbolic solutions.  $a$  will be a list of the  $x$  values and  $b$  will be a list of the  $y$  values. However if we then do

```
>> h = cos(x+y)-2^x+3
>> [a b]=solve(f,h)
```

Matlab won't be able to solve the equations  $f = 0$ ,  $h = 0$  exactly and will return two numerical solutions.

Note: when using the `solve` command matlab will list variables in alphabetical order. Since the variables are named  $x$  and  $y$ , the first letter  $a$  becomes  $x$  and the second letter  $b$  becomes  $y$ . Thus in the examples given below for max-min problems, it is important to name the variables and results as given.

#### Classifying Critical Points

Let us see how we would use Matlab to solve the following problem. Find and classify (relative max, relative min, saddle, or degenerate) all the critical points of  $f(x, y) = x^3 - 12x + (x - y)^2 + 7$ .

```
>> syms x y
>> f = x^3-12*x+(x-y)^2+7
>> fx = diff(f,x)
```

```

>> fy = diff(f,y)
>> fxx = diff(fx,x)
>> D = fxx*diff(fy,y) - diff(fx,y)^2
>> [ax ay] = solve(fx,fy)
>> T = [ ax ay subs(D, {x y}, {ax ay}) subs(fxx, {x y}, {ax ay}) ]
>> double(T)

```

The next to last line takes the list of solutions you found in  $ax$  and  $ay$  and makes a table  $T$  with four columns, consisting of  $x$ ,  $y$ ,  $D$ , and  $f_{xx}$ . In some cases the results could be unintelligible, so the last line prints out the table  $T$  as a table of numbers. From this you can easily read off the type of each critical point. My output was

```

[ 2, 2, 24, 14]
[ -2, -2, -24, -10]

```

That is, the two critical points are  $(2, 2)$  and  $(-2, -2)$ . For  $(2, 2)$ ,  $D = 24 > 0$  and  $f_{xx} = 14 > 0$  so  $(2, 2)$  is a local minimum. For  $(-2, -2)$ ,  $D = -24 < 0$  so  $(-2, -2)$  is a saddlepoint.

### Lagrange Multipliers

Now let us see how to solve an optimization problem using Lagrange multipliers. Find the maximum and minimum of the function  $f(x, y, z) = x^3 - 3x^2 + 2y^2 + z^2$  subject to the constraint  $2x^2 + y^2 + z^2 = 1$ . Note that the lines starting with a `%` are comments and need not be typed in.

```

>> syms x y z lam
>> f = x^3-3*x^2+2*y^2+z^2
>> g = 2*x^2+y^2+z^2
>> fx = diff(f,x)
>> fy = diff(f,y)
>> fz = diff(f,z)
>> gx = diff(g,x)
>> gy = diff(g,y)
>> gz = diff(g,z)
>> % First solve the lagrange multiplier problem grad f = lam*grad g, g=1.
>> [alam ax ay az] = solve( fx-lam*gx, fy-lam*gy, fz-lam*gz, g-1 )
>> % For Lagrange multipliers, we must also include points where grad g =0 and g=1
>> [bx by bz] = solve( gx, gy, gz)
>> % Now make a table with all our critical points and the values of g and f.
>> % Make line breaks as indicated so matlab starts the b values on a new line
>> T = [ ax ay az subs(g, {x y z}, {ax ay az}) subs(f, {x y z}, {ax ay az})
      bx by bz subs(g, {x y z}, {bx by bz}) subs(f, {x y z}, {bx by bz}) ]
>> % Print out the table in a more readable form
>> double(T)
>> % Now look at the table and see if we must eliminate some of the rows.
>> % Eliminate any rows not satisfying the constraint g = 1
>> % Also eliminate any rows with genuinely complex values of x, y, or z.

```

```
>> % The following command prints out only rows 1,2,3,4,7,8 and columns 1-3 and 5
```

```
>> double(T([1,2,3,4,7,8],[1,2,3,5]))
```

The output of the above command is

0.7071	0	0	-1.1464
-0.7071	0	0	-1.8536
0	1.0000	0	2.0000
0	-1.0000	0	2.0000
0	0	1.0000	1.0000
0	0	-1.0000	1.0000

so we see the maximum of 2 occurs when  $x = z = 0$ ,  $y = \pm 1$  and the minimum of  $-1.8536$  occurs when  $y = z = 0$ ,  $x = -0.7071$ .

### Warnings

The Matlab `solve` will automatically find complex solutions. For example  $1.23 + 3.784i$ . But don't throw out seemingly complex solutions such as the symbolic solution  $2 + (2^{1/2}) - 2/2^{1/2}i$  or the numeric solution  $3.2 + .0000i$ . These are both likely real since the imaginary part is zero or very nearly so. You can easily check whether a complicated symbolic solution is real by using `double`.

The above examples were done on a particular version of matlab which may differ from your version in some ways. While the commands will work, it is possible that Matlab's output may differ. For example, `solve` may give you the solutions in a different order than that implied by the examples above. For example in the Lagrange multiplier problem, your complex solutions may occur in different rows.

### Matlab Assignment 3, due Friday, April 6

You must provide me with printed copies of your output with your answers clearly indicated. The first page should be a hand written or printed summary of the answers to the problems below. I encourage you to work in groups of two or three, but no more. Be careful. It is easy to mistype a command and that can throw your results off.

**Problem 1:** Use Matlab to find and classify all critical points of the function

$$f(x, y) = y^4 - x^3 + 2xy^2 + x.$$

Your final answer should be a list of all critical points  $(x, y)$  for which  $x$  and  $y$  are real numbers classified as relative maximum, relative minimum, or saddlepoint. If you can't get the table to print out, that is OK as long as you have Matlab compute the critical points, and compute  $D$  and  $f_{xx}$  for each critical point. Then you can summarize the answers by hand.

**Problem 2:** Use Matlab to find the maximum and minimum of  $x^2 + y^2 + yz^2$  subject to the constraint  $xy + z^2 = 1$ . Your final answer should give the absolute maximum and minimum values and the points at which they occur. Again, if you can't get the table to print out this is OK as long as you have Matlab compute the critical points, compute the function value for each critical point. Then you can summarize the answers by hand. Note: Sometimes Matlab has trouble solving systems that are too easy! For example, for this problem, sometimes my version of Matlab can't do `[bx by bz]=solve(gx,gy,gz)` which asks it to solve the system:  $y = 0, x = 0, 2z = 0$ . (Other times, seemingly at random, it finds the solution  $x = 0, y = 0, z = 0$  just fine.) If this happens to you, you can solve this system by hand. Notice that this point does not satisfy  $g(0, 0, 0) = 1$ , so you don't need these critical points. Thus you can leave these out and use the simpler table

```
>> T = [ ax ay az subs(g, {x y z}, {ax ay az}) subs(f, {x y z}, {ax ay az}) ]
```