Numerical Optimization - Handin 2

Martin Simon Haugaard - CDL966

February 19, 2016

2.6

Examine the statement that *All isolated local minimizers are strict.* This basicly is the same as the logical statement that

$$\text{'isolated local minimizers'} \implies \text{'local minimizers are strict'}$$

The contrapositive statement will then be

$$\text{'local minimizers are not strict'} \implies \text{'local minimizers are not isolated'}$$

Examining this last statement, if a local minimizer, x^*, is to be not strict, there must be other minimizers, x, in the same neighborhood $\mathcal{N}$ which are equal or lower than x^*, meaning that $f(x^*) \geq f(x)$ for $x \in \mathcal{N}$.

Now since there are multiple minimizers in the neighborhood, the local minimizers are not isolated. Proving that the contrapositive statement holds true, which in turn proofs our initial statement.

2.8

Imagine there's a function which will give all the global minimizers, such a function will have to uphold the rule, that for any two points, x and y within its domain $\mathcal{S}$

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y), \text{for all } \alpha \in [0,1]. \tag{1}$$

$\mathcal{S}$ being the set of global minimizers, any point x and y will, when computed in f result in the same value (the global minimum). Thus (1) can be reduced to

$$f(\alpha x(1-\alpha)y) = \text{Global Minimum} \tag{2}$$

From (2) it's evident that any values, x or y will result in being a global minimum. Proving that $\mathcal{S}$ is a convex set, as any two points from the set will result in (2) to remain within the set.

2.9

It's stated in the book, that for p_k to be a downhill direction, the angle Θ_k between p_k and Θf_k has $\cos\Theta_k < 0$, so that

$$p_k^T \nabla f_k = \|p_k\|\|\nabla f_k\| \cos\Theta_k < 0 \tag{3}$$

Calculating ∇f

$$\nabla f = \begin{bmatrix} 2(x_1 + x_2^2) \\ 4x_2(x_1 + x_2^2 \end{bmatrix}$$

It's possible to fill in the values for p_k^T and ∇f_k

$$(-1, 1)\begin{pmatrix} 2 \\ 0 \end{pmatrix} = -2$$

Since $\|p_k\|\|\nabla f_k\|$ is always positive, for the result to be -2, $\cos\Theta_k < 0$ meaning p_k is a descent direction.

Next for solving

$$f(x_1, x_2) = (x_1 + x_2^2)^2 \tag{4}$$

for problem (2.10):

$$f(x_k + \alpha p_k) \tag{5}$$

if $p_k = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$ and $x_k = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ then (5) becomes

$$f(x_k + \alpha p_k) = f(\begin{pmatrix} -1\alpha + 1 \\ \alpha + 0 \end{pmatrix}$$

And finally using (4) there's

$$f(x_k + \alpha p_k) = (1 - \alpha + \alpha^2)^2$$

Now, to find the global minimizers, the gradient has to be zero:

$$\frac{\partial f}{\partial \alpha} = 2(2\alpha - 1)(\alpha^2 - \alpha + 1) = 0$$

For which only $\alpha = \frac{1}{2}$ is the only real solution, being either a local maximum or minimum. To figure out which, the Hessian is examined:

$$\frac{\partial^2 f}{\partial^2 \alpha} = 12\alpha^2 - 12\alpha + 6 \xrightarrow{\alpha=\frac{1}{2}} 3 - 6 + 6 = 3$$

Since the hessian is positive, $\alpha = \frac{1}{2}$ is indeed the local minimum.

Programming Exercises

After having implemented the Jacobian calculation for a given Kinematics chain, it's time to evaluate the result it produces:

$$J = \begin{bmatrix} -0.7071 & 0 & 0 \\ -1.7071 & -1.0000 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The Jacobian matrix, J, above, is the result of the chain which starts in $(0,0,0)$, and has joints in $(0,1.0,0)$, $(-0.7071,1.7071,0)$ and ends in $(-1.7071,1.701,0)$ (Figure 1). What the Jacobian tells us, is the remainder transformation from a given joint, to the final position of the chain.

Looking at these three columns in the J matrix, the final column is zero since the transformation needed upon the last point to each the end-effect is zero, as $\overrightarrow{e} = \overrightarrow{p_3}$.

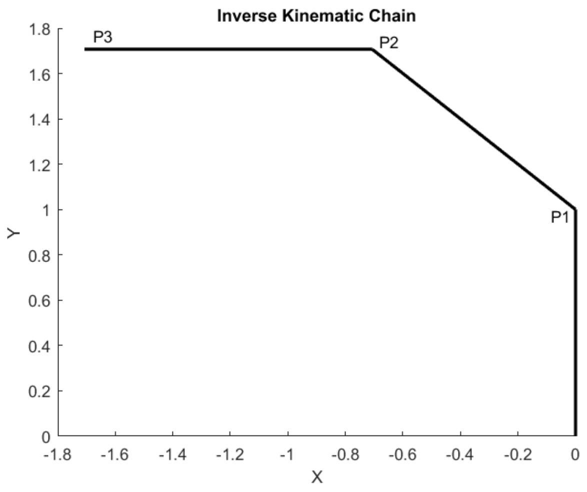

Figure 1: The chain plotted

From P2 the transformation needed is that of moving a distance of 1 along the X-axis. And finally the transformation needed in $P1$ is in both the X (-1.7071) and Y-axis (0.7071, as negative Y-values indicate an increase in value).

Generally speaking, the Jacobian matrix informs us of the distance, and direction a transformation may be needed in order to arrive at the end-effect. And as such, each column in the matrix should ideally reduce the absolute value of the input, until $(0,0,0)$ is achieved, if this was not the case for my input, and it had for example increasingly large values, the Jacobian would be transforming away from the end-effect.

Having re-implemented the Rosenbrock to take a single input of form $z = \begin{bmatrix} x \\ y \end{bmatrix}$, as a script called *rosen*, using the following commands:

```
fun = @rosen;
x0 = [100,100];
options = optimoptions('fminunc','Hessian','on', 'GradObj','on');
[x,g,h] = fminunc(fun,x0,options)
```

fminunc gives the following output:

```
x =   1.0000       1.0000
g =   8.9001e-18
h = 1
```

Once again confirming that $x = (1.0, 1.0)$ is a minimum. Even if the gradient, g, is ever so slightly different from zero, it's well within the uncertainty a optimization algorithm may have.

For good measure I also tried with an initial position of $(-100, -100)$, the gradient remains close to zero. While the Hessian changes to 3, which indicated that the slope on one side of the minimum may be stepper than on the other (as the guesses approach from the opposite site).

It's worth nothing that starting the algorithm with a less accurate guess, by default matlab will stop guessing after 200 iterations, making the initial guess matter beyond reducing the time required for matlab to finish processing.