MINIMIZING THE CARTESIAN FOLIUM

JAN DE LEEUW

ABSTRACT. In this short note we use the Cartesian Folium to illustrate the behaviour of several general purpose minimization algorithms.

1. INTRODUCTION

The "folium cartesii" (letter of Descartes to Mersenne, August 23, 1638) is the function $f : \mathbb{R}^2 \to \mathbb{R}$ defined by

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

The gradient is

$$g(x, y) = \begin{bmatrix} 3x^2 - 3y \\ 3y^2 - 3x \end{bmatrix},$$

and the Hessian is

$$H(x, y) = \begin{bmatrix} 6x & -3 \\ -3 & 6y \end{bmatrix}.$$

It follows that $f(x, y)$ has a saddle point at $(0, 0)$ and an isolated local minimum at $(1, 1)$. These are the only two stationary points. At $(0, 0)$ the eigenvalues of the Hessian are $+3$ and -3, at $(1, 1)$ they are 9 and 3.

The Hessian is singular if and only if (x, y) is on the hyperbola $xy = \frac{1}{4}$. It is positive definite if and only if (x, y) is above the branch of the hyperbola in the positive orthant.

Date: October 27, 2007.

See Figure 1 for contour plots of sections of f on two different scales.

Insert Figure 1 about here

Also, see Figure 2 for the folium with the bowl around $(1,1)$ in the foreground.

Insert Figure 2 about here

2. Newton

In Newton's method we make a quadratic approximation at the current point (x_0, y_0), and then minimize this quadratic approximation to find the new point. The quadratic approximation is

$$f(x,y) \approx f(x_0,y_0)+g(x_0,y_0)'(x-x_0)+\frac{1}{2}(x-x_0)'H(x_0,y_0)(x-x_0).$$

If $H(x_0, y_0)$ is not positive semi-definite, then there is no minimum, and the Newton step is not defined. If $H(x_0, y_0) \trianglerighteq 0$ but singular, then the quadratic approximation has a minimum if and only if $g(x_0, y_0$ is orthogonal the unique vector in the null space of $H(x_0, y_0)$. Some algebra shows that this cannot happen if $x_0, y_0)$ is in the non-negative orthant, and thus the quadratic approximation has a minimum if and only if $H(x_0, y_0) \triangleright 0$.

For $H(x, y)$ positive definite Newton's method uses the algorithmic map

$$F_N(x,y) = \begin{bmatrix} x \\ y \end{bmatrix} - H^{-1}(x,y)g(x,y).$$

By this we mean that in Newton's method we replace $(x^{(k)}, y^{(k)})$ from iteration k by $(x^{(k+1)}, y^{(k+1)}) = F(x^{(k)}, y^{(k)})$. If $H(x, y)$ is not positive definite, then the Newton step is not defined. Making a step only if the Hessian is positive definite is the *supervised* Newton's method. There are other ways to supervise Newton's method, which make it possible to step every time. We will, however, alternatively define *unsupervised* Newton's method by the algorithmic map F_N, where apply F_N whenever $H(x,y)$ is non-singular (or even, using the Moore-Penrose inverse, when $H(x,y)$ is singular). After some algebra the algorithmic map works out as

$$F_N(x,y) = \begin{bmatrix} \frac{2x^2y+y^2}{4xy-1} \\ \frac{2xy^2+x^2}{4xy-1}. \end{bmatrix}$$

The unsupervised Newton's method converges quadratically, either to the saddle point at $(0,0)$ or to the local minimum at $(1,1)$, depending on which side of the hyperbola $xy = \frac{1}{4}$ we choose the starting point of the iterations. Figure 3 colors one million pixels, a pixel is red if Newton's method converges to $(0,0)$ if started from that pixel, and it is yellow if it converges to $(1,1)$. We see that the unsupervised Newton's method converges to the local minimum if and only if the Hessian at the starting point is positive definite.

Insert Figure 3 about here

In Figure 4(a) we have colored the pixels (x,y) for which $F_N(x,y)$ is in the positive quadrant. This is the region above the hyperbola $xy = \frac{1}{4}$. The other plots in Figure 4 make similar plots for the other three quadrants. In each case, one million pixels are colored. They are red if the update of that pixel is in the relevant quadrant.

Insert Figure 4 about here

$$F(x,y) \sim -\begin{bmatrix} y^2 \\ x^2 \end{bmatrix}$$

3. Coordinate Descent

The dynamics of Newton's method are quite complicated. A simpler, but generally slower, algorithm is considered next. In coordinate descent an iteration consists of two parts. We first optimize f

over x, while keeping y fixed at its current value, and then we optimize over y, with x fixed at the new value we have just computed in the new part.

The minimum over x for fixed y only exists if $y > 0$, in which case it is attained at $\sqrt{y}$. In the same way, the minimum over y for fixed $x > 0$ is attained at $\sqrt{x}$. Thus the algorithm is simply

$$x^{(k+1)} = \sqrt{y^{(k)}},$$
$$y^{(k+1)} = \sqrt{x^{(k+1)}},$$

and the algorithmic map is

$$F(x, y) = \begin{bmatrix} \sqrt{y} \\ \sqrt[4]{y} \end{bmatrix}.$$

The algorithm can only work if we start with $y^{(0)} > 0$. It then converges, linearly and monotonically, to $(1, 1)$ with convergence rate $\frac{1}{4}$. The supervised algorithm cannot converge to $(0, 0)$. If we defined, analogously with Newton's method, an unsupervised version that applies the algorithmic map for all $y \geq 0$, then we can indeed have convergence (in a single step) to $(0, 0)$.

4. Quadratic Majorization

As in Newton's method, quadratic majorization methods we make a quadratic approximation at the current point (x_0, y_0), and then minimize this quadratic approximation to find the new point. But we now choose the quadratic approximation in such a way that it is always above the function we are minimizing, while it touches the function in the current point. Thus we want $Q(x_0, y_0)$ such that

$$f(x, y) \leq f(x_0, y_0) + g(x_0, y_0)'(x - x_0) + \frac{1}{2}(x - x_0)'Q(x_0, y_0)(x - x_0).$$

Unfortunately for cubics, such as the Folium, quadratic majorizers do not exist. We use a hack, and minimize the folium on a bounded rectangle.

If we minimize $f(x,y)$ on the rectangle defined by $0 \leq x \leq K$ and $0 \leq y \leq K$ then we can apply quadratic majorization.

$$x^3 \leq x_0^3 + 3x_0^2(x - x_0) + 3K(x - x_0)^2,$$
$$y^3 \leq y_0^3 + 3y_0^2(y - y_0) + 3K(y - y_0)^2,$$

and thus the algorithmic map is

$$F(x,y) = \frac{1}{2K}\begin{bmatrix} -x^2 + 2Kx + y \\ -y^2 + 2Ky + x \end{bmatrix}.$$

The linear convergence rate is $1 - \frac{1}{2K}$

Appendix A. code

```
inequals<-function(x,y,flist)
{
u<-matrix(1,length(x),length(y))
for (k in 1:length(flist))
        u<-u*ifelse(outer(x,y,function(x,y) flist[[k]](x,y))>0,1,0)
image(x,y,u,zlim=c(.5,1),col="RED")
}

attractor<-function(x,y) {
        z<-matrix(0,length(x),length(y))
        for (i in 1:length(x)) {
                for (j in 1:length(y)) {
                        d<-newtfol(x[i],y[j])
                        if (sqrt(sum(d^2))<1e-3) z[i,j]<--1
                        if (sqrt(sum((d-c(1,1))^2))<1e-3) z[i,j]<-1
                }
        }
        image(x,y,z,col=heat.colors(3))
}

newtfol<-function(x,y) {
        xold<-g1(x,y); yold<-g2(x,y)
        repeat {
                xnew<-g1(xold,yold); ynew<-g2(xold,yold)
                if (sqrt((sum((xold-xnew)^2))) < 1e-6) break
                        xold<-xnew; yold<-ynew
        }
return(c(xnew,ynew))
```

```
}

pdf("inequals.pdf")

x<-seq(-3,3,length=1000)
y<-seq(-3,3,length=1000)


f1<-function(x,y) 4*x*y-1
f2<-function(x,y) 2*y*x^2+y^2
f3<-function(x,y) 2*x*y^2+x^2

g1<-function(x,y) f2(x,y)/f1(x,y)
g2<-function(x,y) f3(x,y)/f1(x,y)
g3<-function(x,y) -g1(x,y)
g4<-function(x,y) -g2(x,y)
inequals(x,y,list(g1,g2))
inequals(x,y,list(g1,g4))
inequals(x,y,list(g3,g2))
inequals(x,y,list(g3,g4))

attractor(x,y)

dev.off()
```

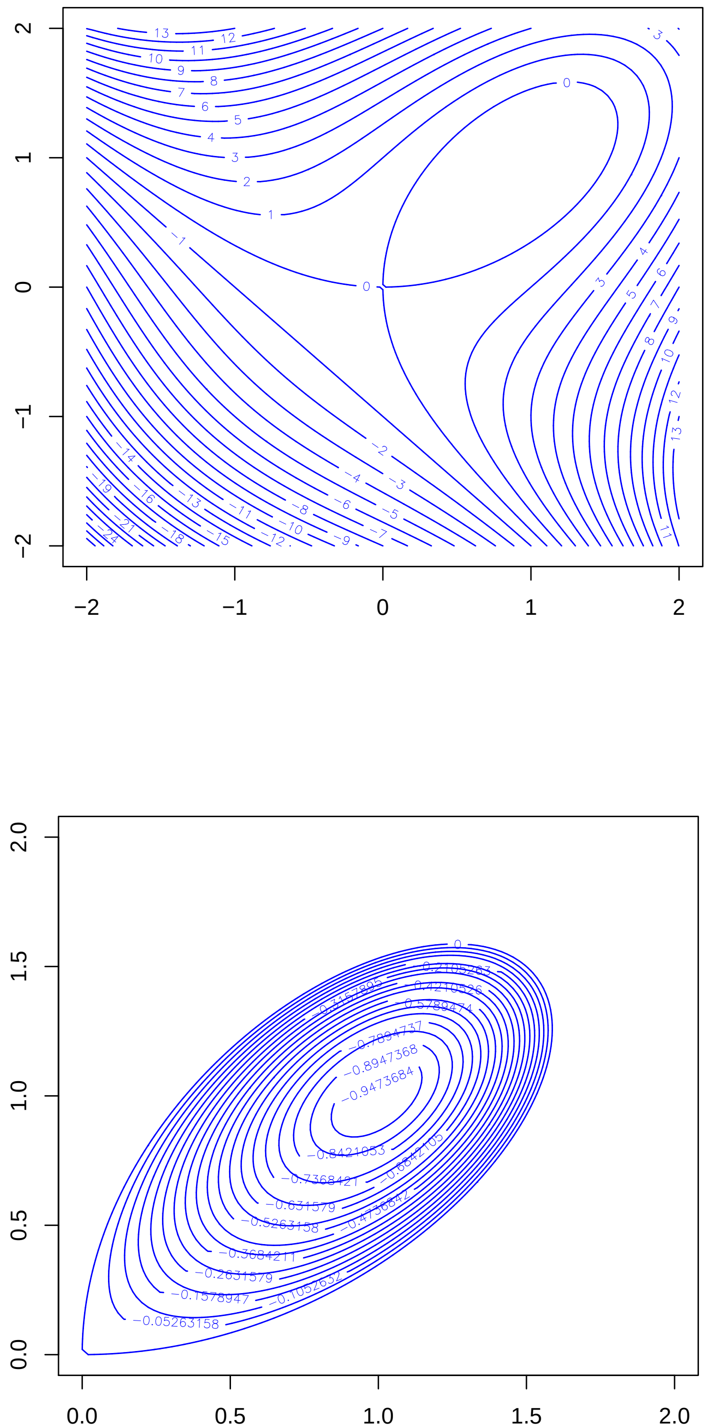


FIGURE 1. Contour Plots for the Folium

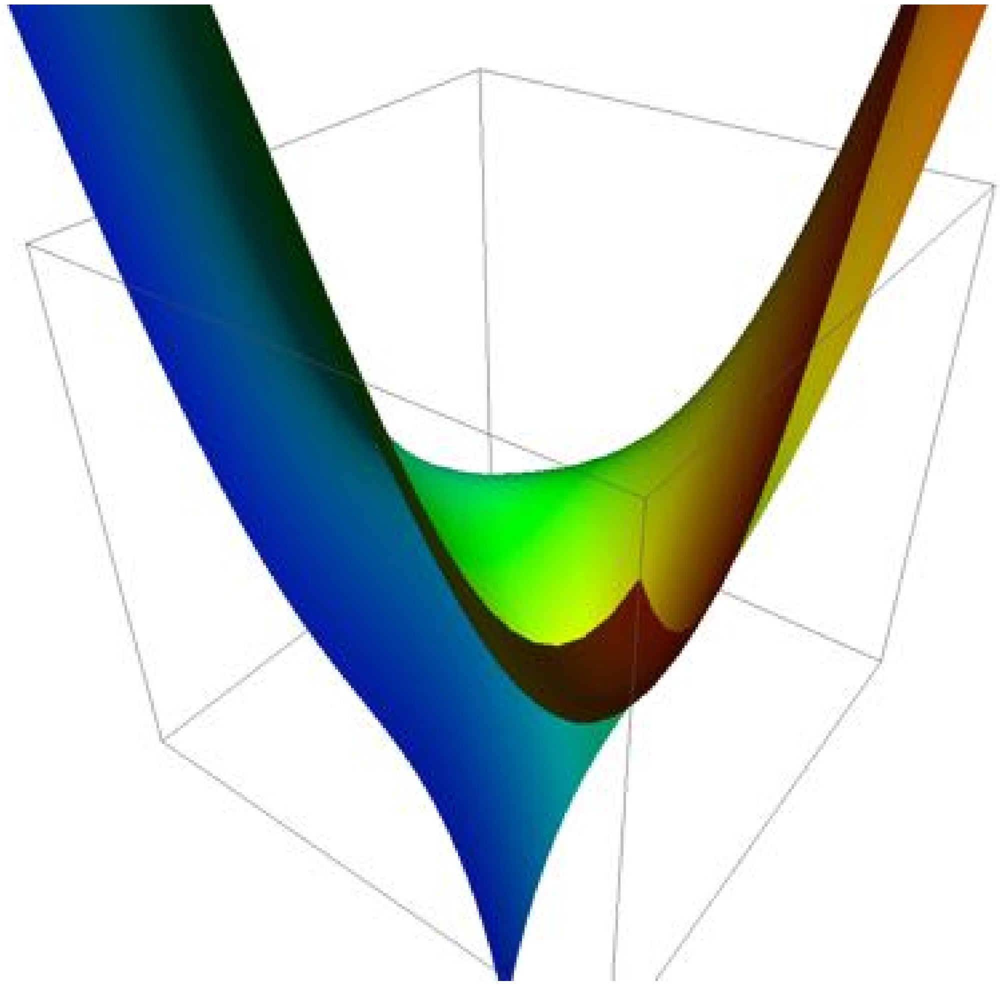


FIGURE 2. The Folium

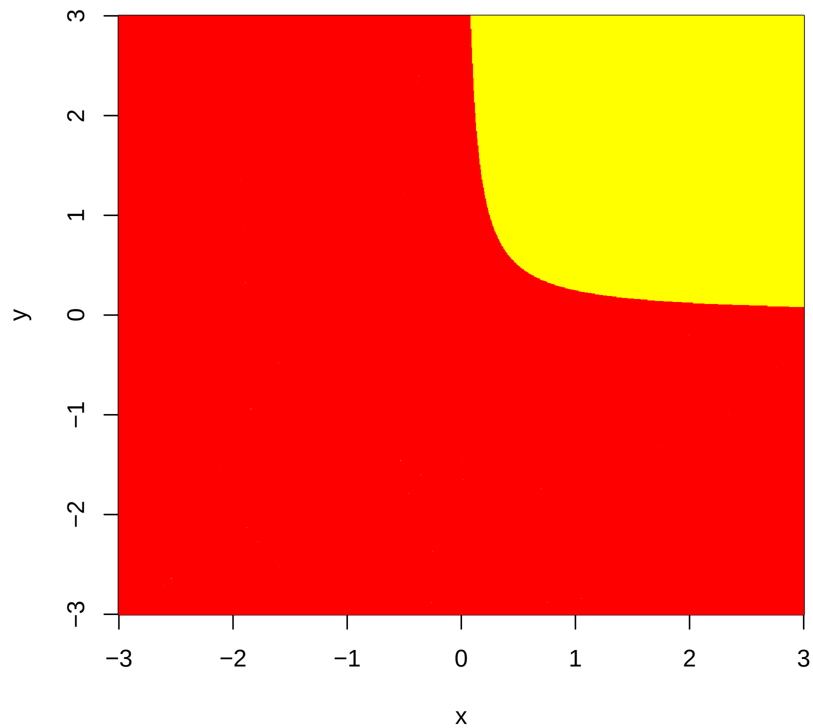
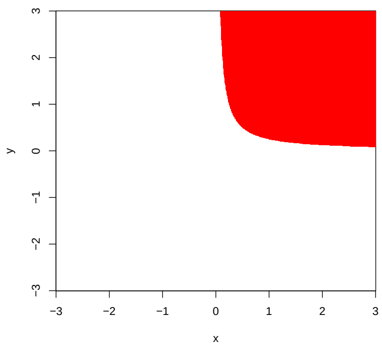
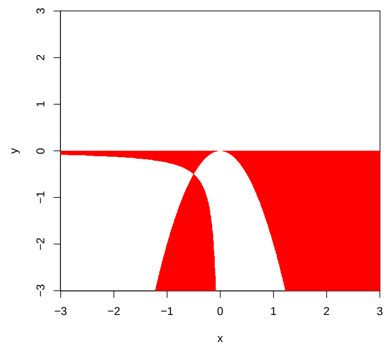


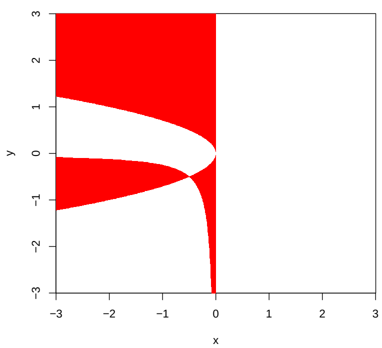
FIGURE 3. Points of Attraction for Newton's Method



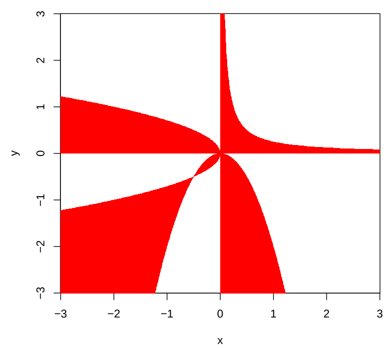
(a) $x > 0$ and $y > 0$



(b) $x > 0$ and $y < 0$



(c) $x < 0$ and $y > 0$



(d) $x < 0$ and $y < 0$

FIGURE 4. Sign of Newton Iterates

Department of Statistics, University of California, Los Angeles, CA 90095-1554

E-mail address, Jan de Leeuw: `deleeuw@stat.ucla.edu`

URL, Jan de Leeuw: `http://gifi.stat.ucla.edu`