

MULTIVARIABLE CALCULUS

OLIVER KNILL, MATH 21A

Lecture 24: The gradient

GRADIENT THEOREM

A most important theorem in multi-variable calculus is the **gradient theorem**:

$\nabla f(x_0, y_0)$ is perpendicular to the level curve passing through (x_0, y_0) .

The proof is that if the level curve is parametrized as $\vec{r}(t)$, then the function does not change so that $0 = \frac{d}{dt}f(\vec{r}(t)) = \nabla f(\vec{r}(t)) \cdot \vec{r}'(t)$ which means that the gradient $\nabla f(\vec{r}(t))$ is perpendicular to the velocity $\vec{r}'(t)$. In higher dimensions, the proof is similar, just check it for every curve.

ANGLES DANCE UPWARDS

A most important paradigm in **machine learning** is that we can improve a function by going into the direction of the gradient. This is the **angel theorem** because angels dance upwards:

The function f increases most in the gradient direction.

The proof also follows from the chain rule. We know

$$\frac{d}{dt}f(\vec{r}(t)) = \nabla f(\vec{r}(t)) \cdot \vec{r}'(t) = |\nabla f(\vec{r}(t))| |\vec{r}'(t)| \cos(\alpha).$$

This is biggest if $\alpha = 0$. This means if $\vec{r}'(t)$ points into the same direction than ∇f .

QUADRATIC APPROXIMATION

We have seen the linear approximation $L(x, y) = f(x_0, y_0) + f_x(x_0, y_0)(x - x_0) + f_y(x_0, y_0)(y - y_0)$. The next best improvement is the quadratic approximation. In one dimensions it is

$$Q(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)(x - x_0)^2}{2}.$$

In two dimensions, all the 4 derivatives $f_{xx}, f_{xy}, f_{yx}, f_{yy}$ appear. Because of Clairaut $f_{xy} = f_{yx}$, we have

$$Q(x, y) = L(x, y) + \frac{[f_{xx}(x_0, y_0)(x - x_0)^2 + 2f_{xy}(x_0, y_0)(x - x_0)(y - y_0) + f_{yy}(x_0, y_0)(y - y_0)^2]}{2}.$$

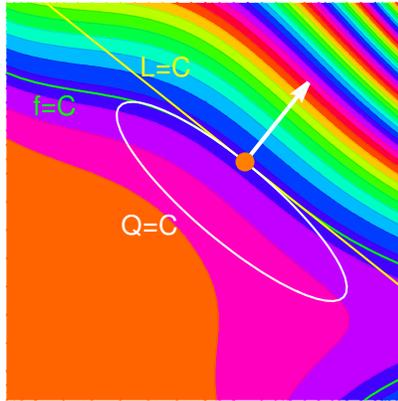


FIGURE 1. The linear and quadratic approximations $L(x, y)$ and $Q(x, y)$ in the contour picture. We see the contours of f and the contour of L and Q passing through the points. We see also that the gradient is perpendicular to all these three curves at (x_0, y_0) .

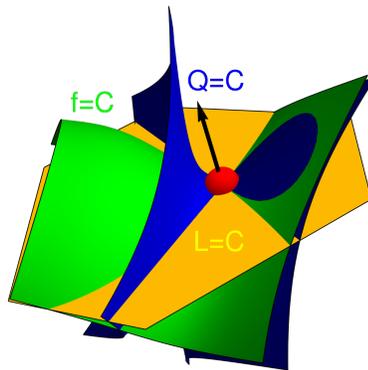
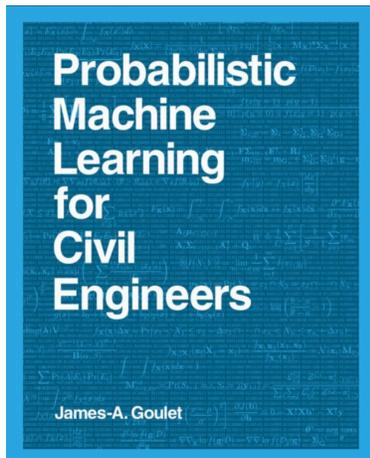


FIGURE 2. The linear and quadratic approximations $L(x, y)$ and $Q(x, y)$ are seen as graphs. They are both tangent to the graph.



PROBABILISTIC MACHINE LEARNING FOR CIVIL ENGINEERS 49

Algorithm 1: Gradient ascent with backtracking line search

```

1 initialize  $\lambda = \lambda_0$ ,  $\theta_{k+1} = \theta_k$ , define  $e, \epsilon, \beta$ 
2 while  $|f(\theta_{k+1}) - f(\theta_k)| > \epsilon$  do
3   compute  $\left\{ \begin{array}{l} f(\theta_{k+1}) \\ \nabla f(\theta_{k+1}) \end{array} \right.$  (Function value)
4   compute  $\theta_{k+2} = \theta_{k+1} + \lambda \nabla f(\theta_{k+1})$  (1st derivative)
5   if  $f(\theta_{k+2}) < f(\theta_{k+1}) + \lambda \nabla f(\theta_{k+1})^T (\theta_{k+2} - \theta_{k+1})$  then (Backtracking)
6     set  $\lambda = \lambda/2$ 
7   until  $\lambda = \lambda_0$ ,  $\theta_{k+2} = \theta_{k+1}$ 
8   set  $\theta^* = \theta_{k+1}$ 

```

Figure 3.1 presents the first two steps of the application of algorithm 1 to a non-convex/non-monotone function with an initial value $\theta_0 = 3.5$ and a scaling factor $\lambda_0 = 3$. For the second step, the scaling factor λ has to be reduced twice in order to satisfy the Armijo rule. One of the difficulties with gradient ascent is that the convergence speed depends on the choice of λ_0 . If λ_0 is too small, several steps will be wasted and convergence will be slow. If λ_0 is too large, the algorithm may not converge.

Figure 3.2 presents a limitation common to all convex optimization methods when applied to functions involving local maxima: if the starting location θ_0 is not located on the slope segment leading to the global maximum, the algorithm will most likely miss it and converge to a local maximum. The task of selecting a proper value θ_0 is nontrivial because in most cases, it is not possible to visualize $f(\theta)$. This issue can be tackled by attempting multiple starting locations θ_0 , and by using domain knowledge to identify proper starting locations.

Gradient ascent can be applied to search for the maximum of a multivariate function by replacing the univariate derivative by the gradient so that

$$\theta_{k+1} = \theta_k + \lambda \nabla_x f(\theta_k).$$

As illustrated in figure 3.3, because gradient ascent follows the direction where the gradient is maximal, it often displays an oscillatory pattern. This issue can be mitigated by introducing a momentum term in the calculation of θ_{k+1}

$$\nu_{k+1} = \gamma \nu_k + \lambda \nabla_x f(\theta_k),$$

$$\theta_{k+1} = \theta_k + \nu_{k+1}$$

where ν can be interpreted as a velocity that carries the momentum from the previous iterations.

Figure 3.4: Example of application of gradient ascent with backtracking for finding the maximum of a function.

Figure 3.5: Example of application of gradient ascent with backtracking for finding the maximum of a function.

Figure 3.6: Comparison of gradient ascent with and without momentum.

Hoschek, D. R., G. E. Hinton, and R. J. Williams (1988). Learning representations by back-propagating errors. *Neural Netw.* 1(5): 305-306.

FIGURE 3. The gradient is important in machine learning. It can be used there to learn. Learn it to use it!