Regression with Gradient Descent

October 1, 2015

1

First, we discuss variations on gradient descent, including analytic gradient descent, finite difference gradient descent, and Matlab's `fminunc`, which includes some variations such as adaptive step size, using a quadratic approximation instead of linear approximation, and using a 2-dimensional subspace to reduce computational complexity. First, we'll examine the analytic gradient descent, which relies on having an analytic gradient expression at all points in space.

We'll refer to three example functions: the n-dimensional quadratic "bowl", Q_n, the n-dimensional inverted Gaussian (centered at $\mu = 0$, with $\Sigma = \mathbf{I}_n$), N_n, and the n-dimensional sum of sin's, S_n. These are defined as follows (leaving off the normalization constant on the inverted Gaussian for simplicity):

$$Q_n = \|\mathbf{x}\|^2$$

$$N_n = -\exp\left(-\frac{1}{2}\|\mathbf{x}\|^2\right)$$

$$S_n = \sum_{i=1}^{n} \sin(x_n)$$

Next, we benchmark the gradient descent variations on these two functions (seeding with an initial guess of $(1, 1, \ldots)$):

Table 1: Iterations (step = .1, threshold = .001)

	Analytic Gradient			Finite Differences			`fminunc`		
n	Q_n	N_n	S_n	Q_n	N_n	S_n	Q_n	N_n	S_n
2	16	33	46	16	33	46	6	15	18
5	18	60	50	18	60	50	12	48	36
20	21	1	57	21	1	57	42	21	126
100	25	1	64	25	1	64	202	101	606

Focusing on just S_n, and leaving the step size, starting point, and threshold the same unless otherwise noted, we may also examine the effects of each variable (defaulting to $n = 5$:

Table 2: Examining the effect of other variables

Step Size		Starting Point		Threshold	
Δ	Iterations	x_0	Iterations	δ	Iterations
2.0	177	0.001	49	0.01	39
1.0	5	0.01	49	0.001	50
0.1	61	0.1	50	0.0001	61
0.01	503	1.0	61	0.00001	72

We notice that the finite differences and analytic methods yield the exact same number of iterations for all functions. This can be explained by examining the diagonal of the Hessian matrix for

each of the functions - in all cases, the values are small, indicating low curvature, and thus that the function can be well-approximated by the linear finite difference approximation. Therefore, when examining the effects of other variables, we only present the numbers using the finite-difference gradient (the analytic gradient produces identical numbers). We may draw a few conclusions by examining the effects of other variables. As we decrease the step size, it takes longer to converge. However, increasing the step size beyond 1 causes the iteration to overshoot, resulting in an overall higher number of required iterations. Moving the starting point further from the minimum increases the number of iterations, but only slightly, because the gradient increases, which increases the step size proportionally. Finally, requiring a stricter convergence criteria via a smaller threshold increases the number of iterations, but only by a constant amount, because the gradient again ensures the step size is in proportion to the distance from the minimum.

By plotting the difference between the finite differences approximation, we can see the accuracy achieved. Here, we again use $\delta = .01$. Figures 1 through 3 are plots of $(\nabla f(\mathbf{x}) - \tilde{\nabla} f(\mathbf{x}))/f(\mathbf{x})$ for $x = (0, 1)$.

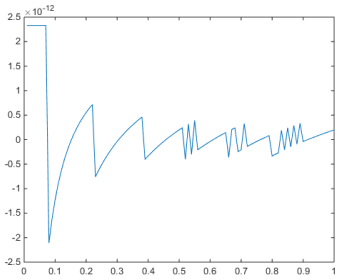

Figure 1: Q_1, scale of 10^{-12}

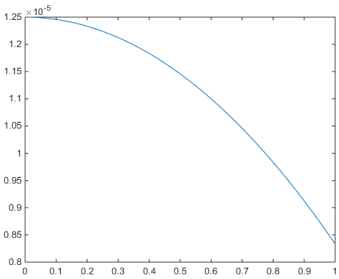

Figure 2: N_1, scale of 10^{-5}

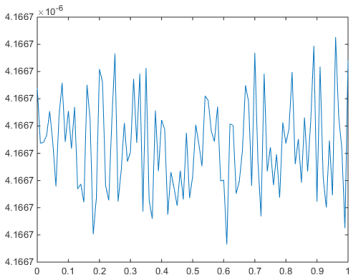

Figure 3: S_1, scale of 10^{-6}

Given the scales of the plots, it is clear that the finite differences method is quite accurate for these functions over these distance scales. As noted, this is generally true when the diagonal entries of the Hessian matrix are small.

2

3

Here we examine ridge regression, which is a form of regression that attempts to minimize the coefficients of Θ. It does this by imposing a penalty, λ, on the norm of the coefficients of Θ. The analytic solution to this minimization is given by

$$\theta = (\Phi^\intercal \Phi + \lambda \mathbf{I})^{-1} \Phi^\intercal y$$

Where $\Phi = \phi(x)$. We then have the predictor function given by ϕ weighted by θ, eg, we would predict $y_{predicted} = \theta^\intercal \phi(x')$. We can see the effect λ has on the result by examining a few values of λ for the same dataset used previously (generated from $\sin(2\pi x)$). The effect of λ is clear in Figures 4 and 5, where higher values of λ lead to flatter, smoother approximations. As M increases (especially relative to n), the values in θ begin rising enormously, but adding the λ factor helps to keep them under control. For example, in this example with $M = 7$, $\max \theta_i = 1595.9$ for $\lambda = 0$, but for $\lambda = .01$, $\max \theta_i = 1.7781$, a much more reasonable value, and far more likely to approximate the truth accurately.

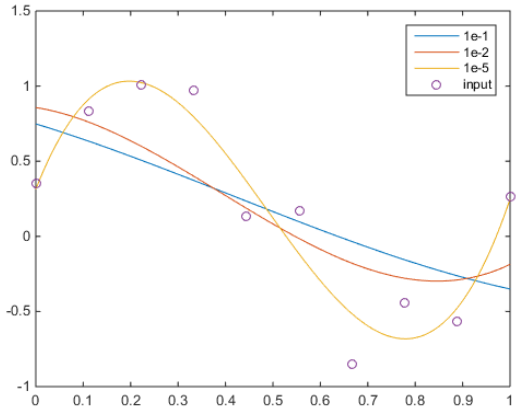

Figure 4: $M = 3$

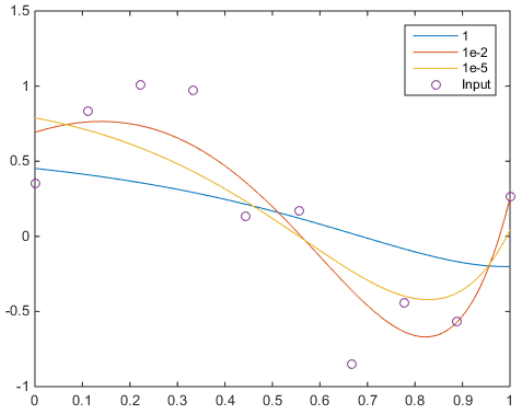

Figure 5: $M = 7$