# 1 Gradient Descent

## 1.1 Batch Gradient Descent on Gaussian and Quadratic Objective Functions

We implemented a basic batch gradient descent procedure for minimizing functions and tested this procedure on a Gaussian and a quadratic function (Figure 1). These functions have known minima at (10,10) and (26.67, 26.67) respectively. This procedure takes in a constant step size, a starting guess vector, and a criteria for convergence. We examined the effects of these parameters.

**Step Size**: For both functions, decreasing the step size caused the gradient descent procedure to converge slower (Fig 1a,e). This makes sense as a smaller step size will cause a smaller update in each iteration.

**Starting Guess**: The effect of the starting guess varied between the Gaussian and Quadratic functions (Fig 1b,f). For the Gaussian function, whose minimum is at (10,10), the number of iterations to convergence increases as the Euclidean distance between the starting guess and true minimum increases. Furthermore since starting guesses (5,15) and (15,5) are the same distance from the minimum, they converge at exactly the same rate. On the other hand, the time to convergence of the quadratic function generally increases with the distance from the starting guess to the minimum, although not in all cases. Since the covariance of the two inputs is non-zero, the function produces different results for (16.67, 36.67) and (16.67, 16.67) which have the same Euclidean distance from the minimum at (26.67,26.67). When starting at (16.67,16.67) the function starts off higher but converges faster than when starting at (16.67,36.67). The convergence is slower at the second point because the covariation term will make it hard to increase the first variable while decreasing the second variable.

**Convergence criteria**: For both functions there is an inverse relationship between the number of iterations to convergence and the $\log_{10}$ of the convergence criteria (Fig 1c,g). This makes sense as decreasing the convergence criteria by a factor of 10 will mean that the gradient descent procedure must get closer to the true minimum before terminating

Furthermore, for both functions we examined the norm of the gradient (Fig 1d,h). In both cases, the norm of the gradient monotonically decreases and goes close to 0 at the time of convergence.

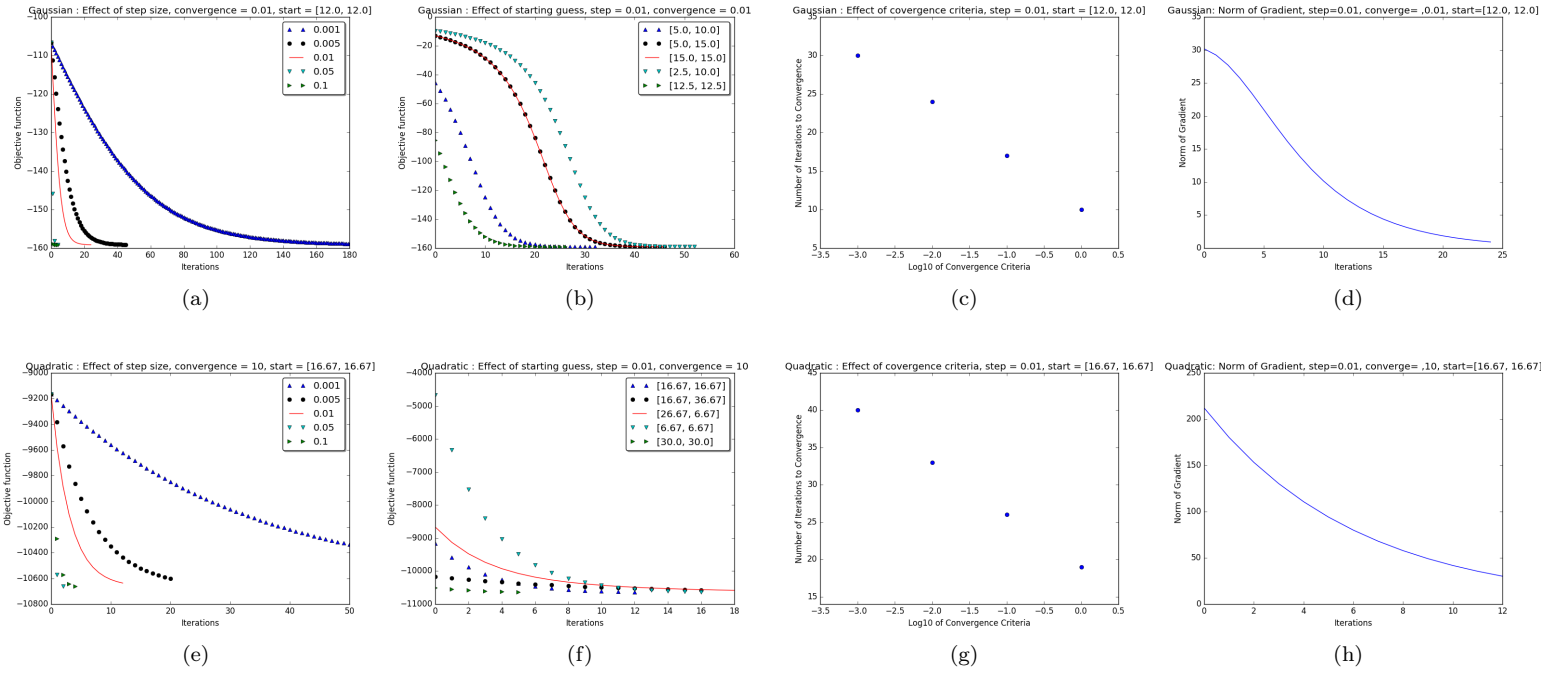



Figure 1: Analysis of gradient descent on Gaussian and Quadratic function.

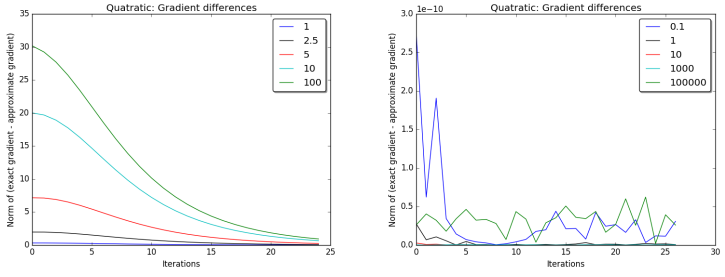



Figure 2: The norm of the difference between the exact gradient and the central difference approximate gradient evaluated as the gradient descent procedure minimizes an objective function. Varied over different step sizes for the approximation. (A) Gaussian function with parameters from Fig 1d and (B) Quadratic function with parameters from Fig 1h

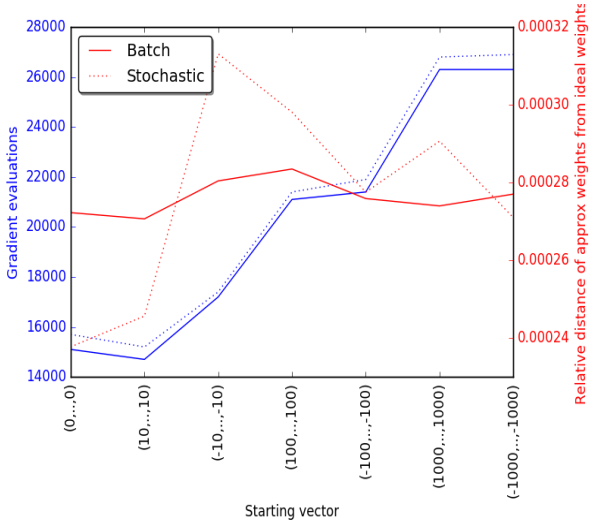



Figure 3: : Comparing batch and stochastic gradient descent for minimizing least squares objective. The number of gradient evaluations (blue) and the relative distance of the calculated weights from the ideal weights (red) for both batch gradient descent (solid line) and stochastic gradient descent (dotted line) at a variety of starting points.

### 1.2 Central Difference Approximation

We used the central difference approximation to evaluate the value of the gradient during the course of a gradient descent procedure. Figure 2 displays the norm of the difference between the exact gradient and the approximate gradient over the course of gradient descent on the Gaussian and Quadratic functions. For the Gaussian function, the approximate gradient is much further from the exact gradient than is the case for the Quadratic function. Note that the y-axis of Figure 2a is on the scale of $10^1$ while the y-axis of Figure 2b is on the scale of $10^{-10}$. Over this range of approximation step sizes, the Gaussian approximate gradient becomes more similar to the Gaussian gradient as the step sizes decreases. On the other hand, the Quadratic approximate gradient is most similar to the Quadratic exact gradient at a step size in the 1 to 10 range. At step sizes higher and lower the difference is greater. This is likely due to the fact that in the central difference approximation we move away from a point by the step size and divide the approximation by the step size. Thus too large of step size will move the approximation too far away, causing inaccuracy, while too small of a step size will produce a large value when the step size is divided, also creating inaccuracy. Thus there exists a sweet spot for the step size, where it is neither too large nor too small and produces the best results.

### 1.3 Stochastic Gradient Descent

In addition to our batch gradient descent procedure, we implemented an alternative procedure using stochastic gradient descent. In this procedure, the learning rate at time $t$ is given by $\eta = (10^8 + t)^{-0.75}$. In Figure 3, we compare batch and stochastic gradient descent when minimizing least squares data. For all starting points, stochastic gradient descent requires slightly more evaluations of the gradient than batch gradient descent does, although this difference is small relative to the total number of evaluations carried out. Furthermore, batch gradient descent produces a weight vector whose relative distance from the ideal weight vector varies only slightly at the different start points. On the other hand, the relative distance of the weight vector with stochastic gradient descent varies much more so and is sometimes larger than that of batch gradient descent and sometimes smaller.

## 2 Ridge Regression

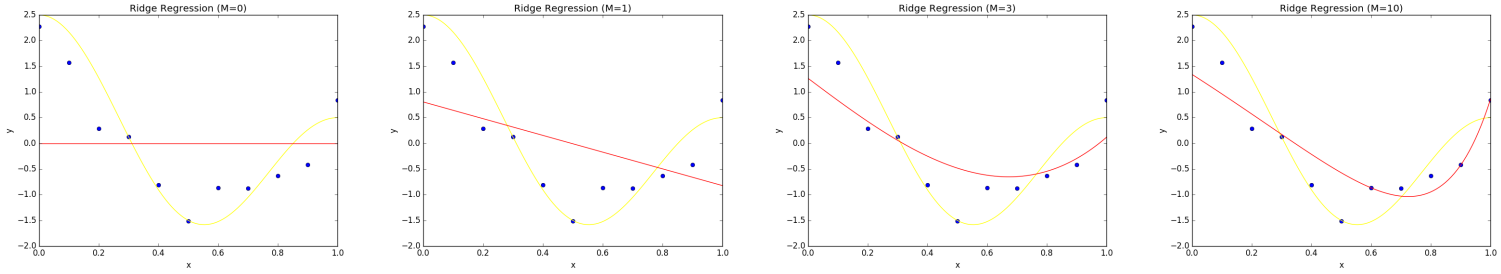
### 2.1 Implementing Ridge Regression

We implemented ridge regression with a polynomial basis function on the curve fitting data from section 2 (Figure 4). As the value of $M$ increases the functions curvature increases as expected. However, as $\lambda$ increases the curvature decreases. Thus, when $M = 10$ and $\lambda = 1$ the curvature is less than when $M = 3$ and $\lambda = 0.1$.

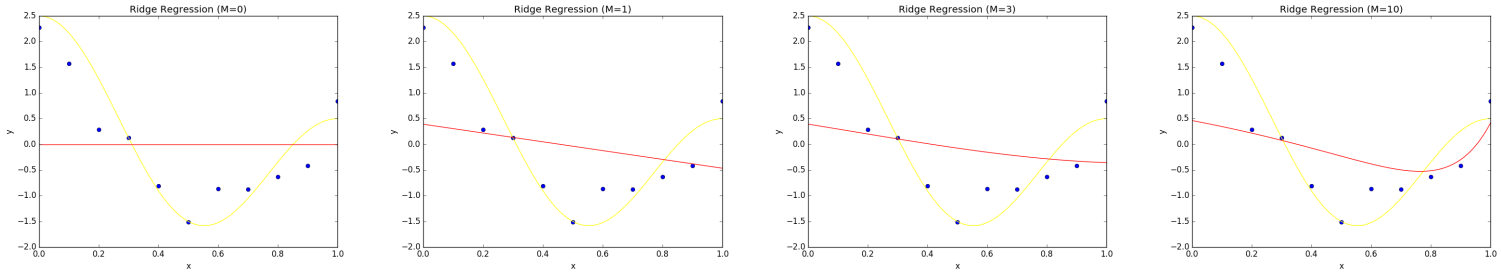
### 2.2 Ridge Regression Validation

We ran our ridge regression with a polynomial basis function on three datasets A, B and validation and tested the performance, as measured by $R^2$, in the following cases: training and regressing on A, training on A and regressing on B, training on A and regressing on validation, training and regressing on B, training on B and regressing on A, training on B and regressing on validation (Figure 5). This was repeated for all parameter combinations of $M \in \{3, 5, 50\}$ and $\lambda \in \{0.1, 10, 100\}$. From the bottom two rows of the table, the highly negative $R^2$ values for $m = 50$ make it clear than ridge regression heavily punishes model that severely overfit the data. Much better performance is seen with much more reasonable $M$ values such as 3 and 5.

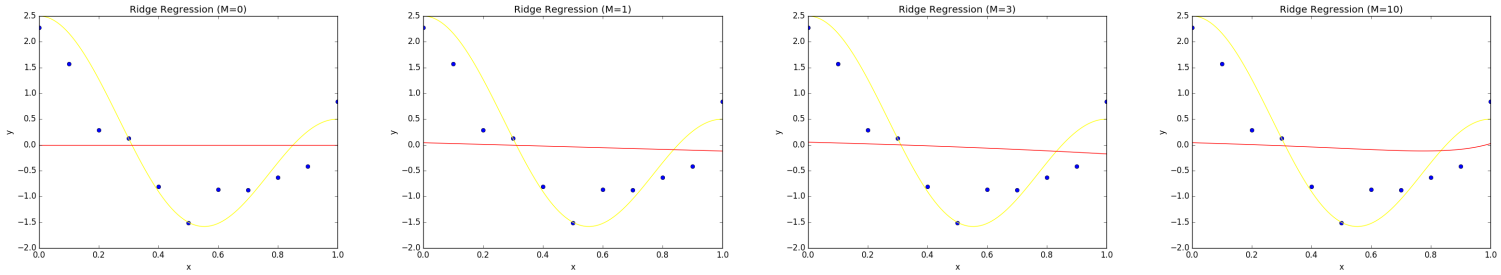
However, it is also clear that there is danger in choosing a $\lambda$ value that is too high. From the top four rows of the table we can see that when $\lambda = 10$ the model with $M = 5$ outperforms the model with $M = 3$ on the validation data. On the other hand, when $\lambda = 100$ the model with $M = 3$ is better than the one with $M = 5$. However, this model is still performs worse on the validation set than both of the models generated with $\lambda = 10$. This illustrates that while increasing $\lambda$ is good for preventing overfitting, doing so runs the risk of generating a model with little predictive power.




(a) $M = 0, \lambda = 0.1$ (b) $M = 1, \lambda = 0.1$ (c) $M = 3, \lambda = 0.1$ (d) $M = 10, \lambda = 0.1$




(e) $M = 0, \lambda = 1$ (f) $M = 1, \lambda = 1$ (g) $M = 3, \lambda = 1$ (h) $M = 10, \lambda = 1$




(i) $M = 0, \lambda = 10$ (j) $M = 1, \lambda = 10$ (k) $M = 3, \lambda = 10$ (l) $M = 3, \lambda = 10$

Figure 4: Ridge regression on curve fitting data from section 2 with polynomial basis function with varying values of $M$, the degree of the polynomial basis, and $\lambda$, the ridge regression parameter

| M, Lambda | Train A | Train A/Test B | Train A/Validate | Train B | Train B/Test A | Train B/Validate |
|---|---|---|---|---|---|---|
| [3, 10] | 0.87 | -0.86 | 0.73 | 0.13 | 0.09 | 0.26 |
| [3, 100] | 0.73 | -1.39 | 0.51 | -0.63 | 0.11 | 0.09 |
| [5, 10] | 0.91 | -0.83 | 0.81 | 0.37 | -1.41 | 0.34 |
| [5, 100] | 0.7 | -1.41 | 0.43 | -0.37 | -0.83 | -0.48 |
| [50, 10] | -25479.24 | -295921712.36 | -2442487509.33 | -1381051.37 | -5.82115310675e+19 | -3.68597308383e+17 |
| [50, 100] | -147437.16 | -27523188280.9 | -8989891448.06 | -413609.64 | -7.98496399779e+18 | -4.71405829654e+16 |

Figure 5: : $R^2$ values for regression on datasets A, B, and validation in the cases of training and regressing on A, training on A and regressing on B, training on A and regressing on validation, training and regressing on B, training on B and regressing on A, training on B and regressing on validation.