Ridge regression

Daniel Eberharter
Claudio Canella

June 4, 2014

1 Algorithmic choices

Our method does not substantially differ from the one discussed in class. We chose two non-linear functions, one is $f(x) = x^2 + 2x + 2$ and the other one is $f(x) = 6x^2+2x+2$. We then compute $r^t(x) = f(x^t)+\epsilon$, where ϵ is drawn from a standard uniform distribution(in octave calculated by stdnormal_pdf) with a random error selected between 0 and 15 weighted by the variance(for full information on calculation please see code). With this function we calculate our training- and validation set. We calculate our design matrix and use the formula that has been introduced in the lecture to calculate our regression coefficients for each λ and to avoid biasing towards w_0 we center our values. By using M-fold cross-validation we find our best λ.

2 numbers and illustrations

- x = 0 to 9
- N = 20
- M = 3
- probability of error: 0.15 (used for ϵ)
- $\lambda = 19$ generated logarithmic values between 0 and 10^6

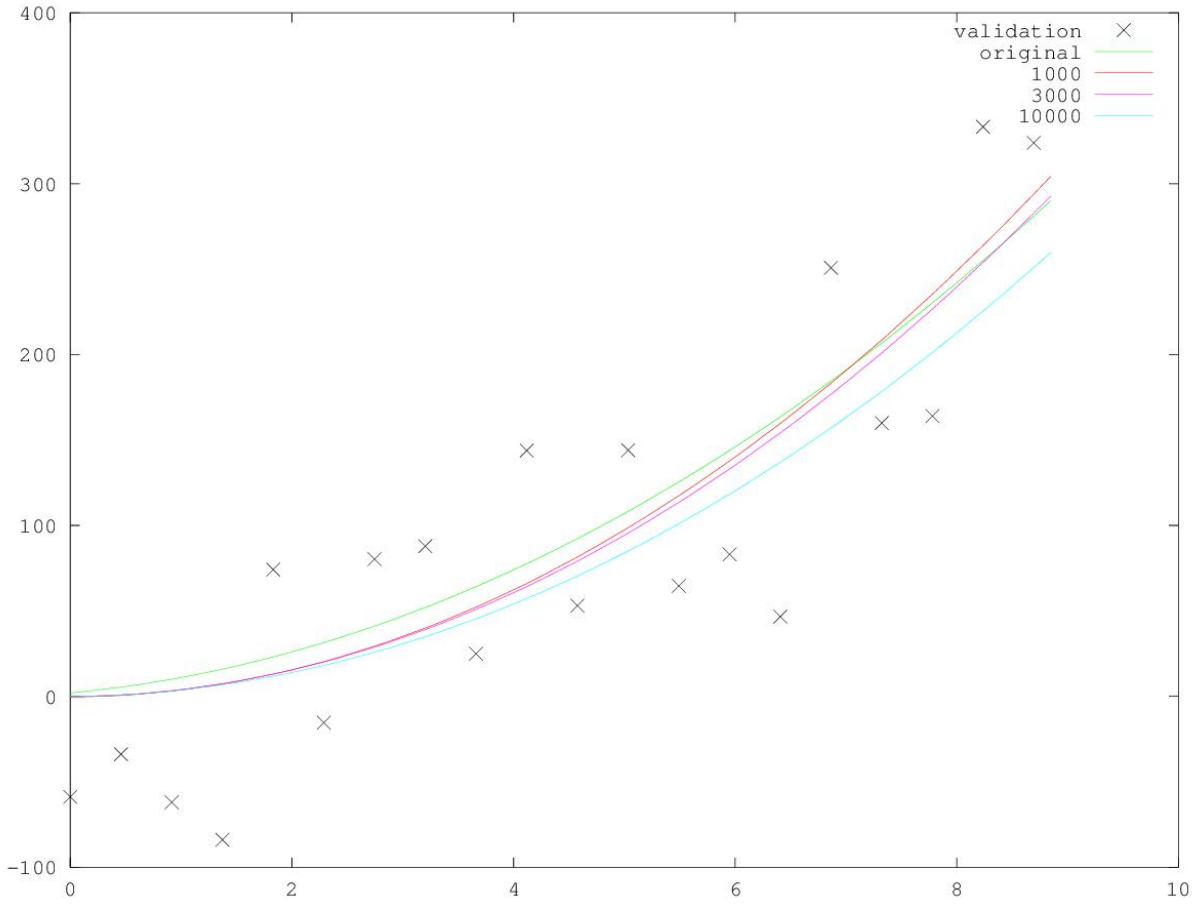

Figure 1: plotted graph with λ values specified as in list 2

- λ plotted = least error, index of lambda at position $1 + (index\_best - \frac{num\_lambda}{2})\%\ num\_lambda$, index of lambda at position $num\_lambda - (1 + (index\_best - \frac{num\_lambda}{2})\%\ num\_lambda)$

As it is part of the task, we created a plot with λ on the abscissa and the error on the ordinateas it can be seen in figures 4 and 5.

In figures we can see all the calculated errors for both functions with the best one being indicated as it is written to the console.

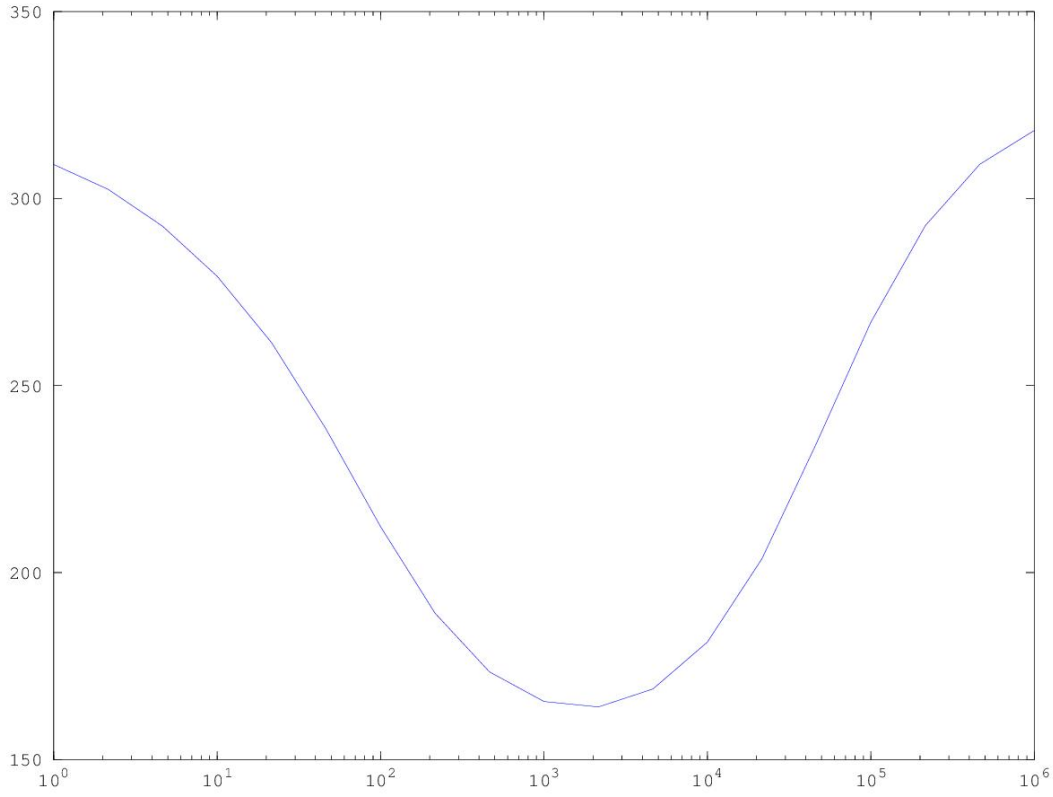

Figure 2: λ and the error plotted on a logarithmic abscissa for $f(x) = 3x^2 + 6x + 2$

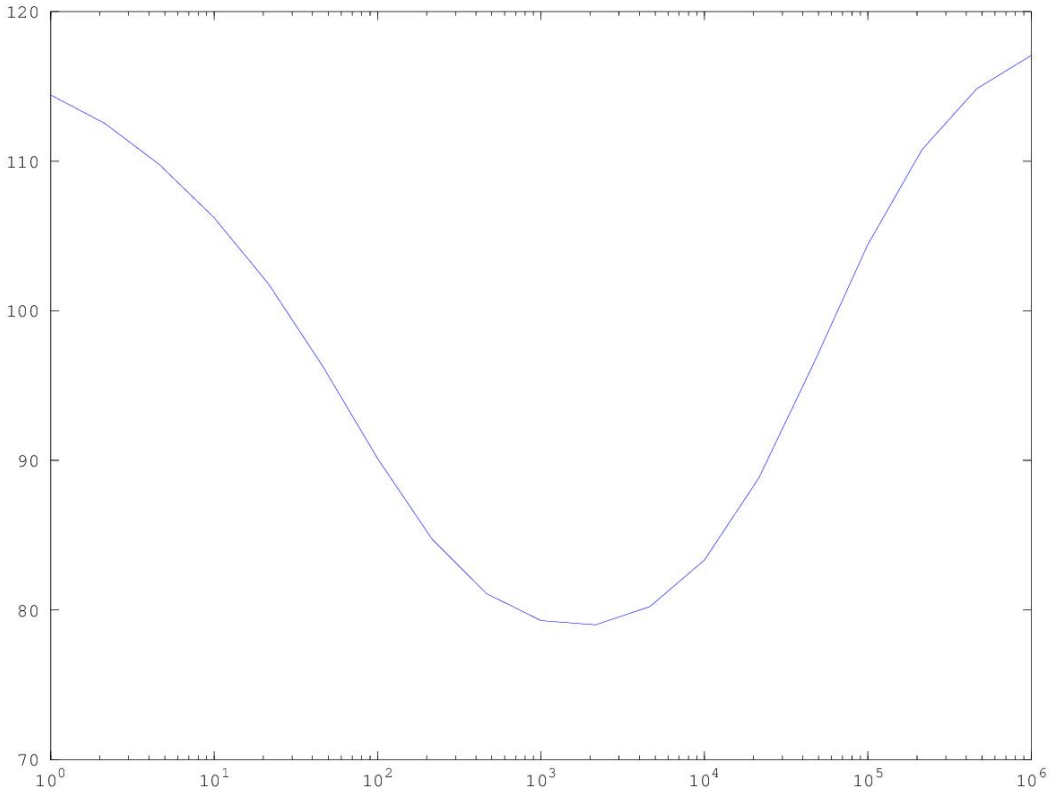

Figure 3: λ and the error plotted on a logarithmic abscissa for $f(x) = x^2 + 2x + 2$

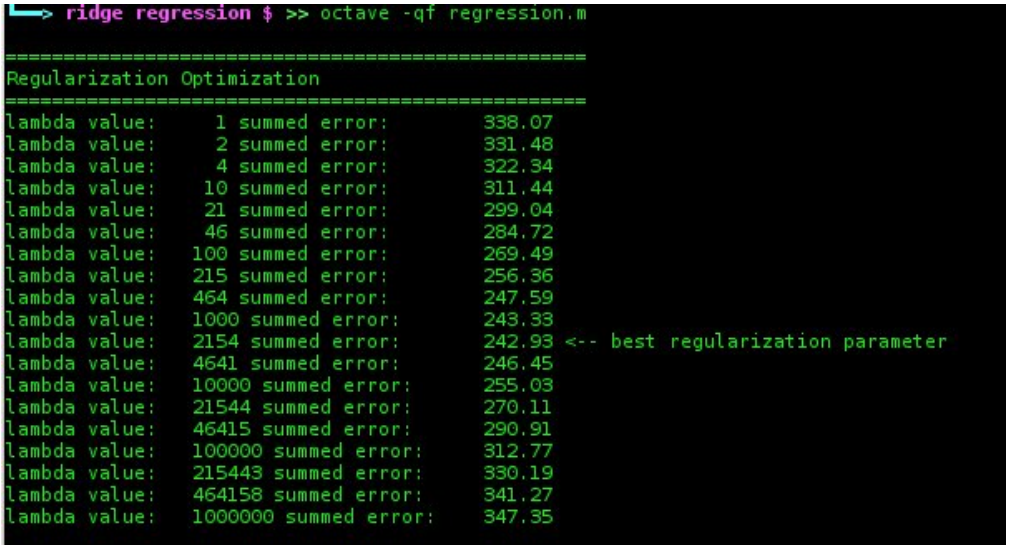

Figure 4: λ and the error for $f(x) = 3x^2 + 6x + 2$

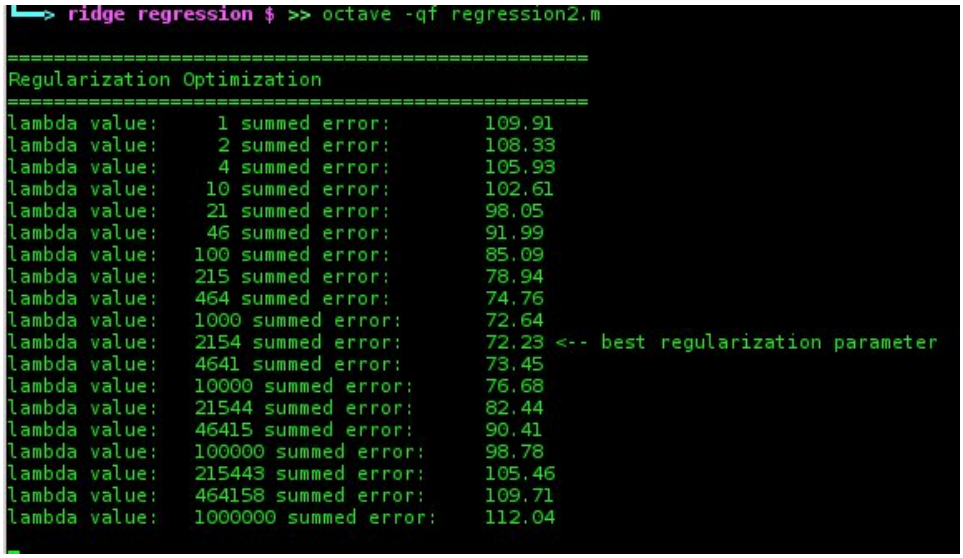

Figure 5: λ and the error for $f(x) = x^2 + 2x + 2$

3 Conclusions

We tested our implementation with different functions without increasing the order of our polynomial as we would have to change the whole implementation of the program. Instead we increased and decreased the weighting factor of each order and our λ values. While doing this we noticed that for the function with the higher slope($f(x) = 3x^2 + 6x + 2$) our approximation with the different λ values is almost identical at the start, but at the end the diverge as seen in figure 6. When we use the function with the lower slope($f(x) = x^2 + 2x + 2$) it is the other way around, meaning that the approximation with the different λ values diverge at the beginning and converge at the end as seen in figure 7.

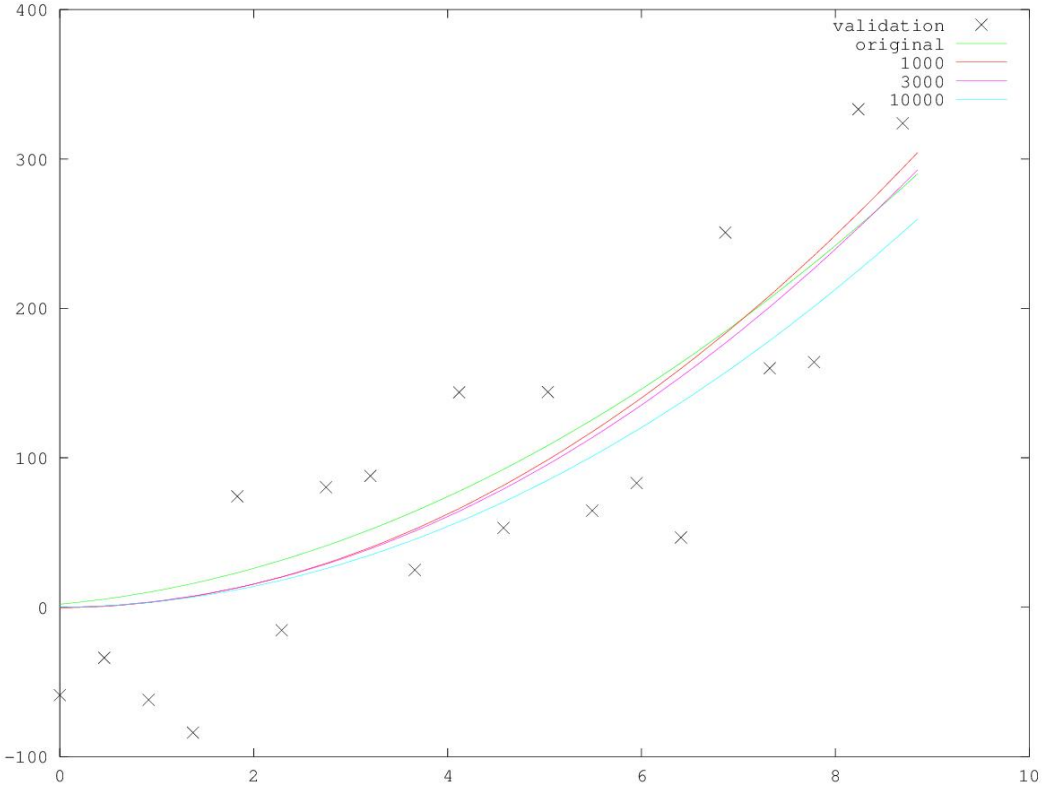

Figure 6: $f(x) = 3x^2 + 6x + 2$

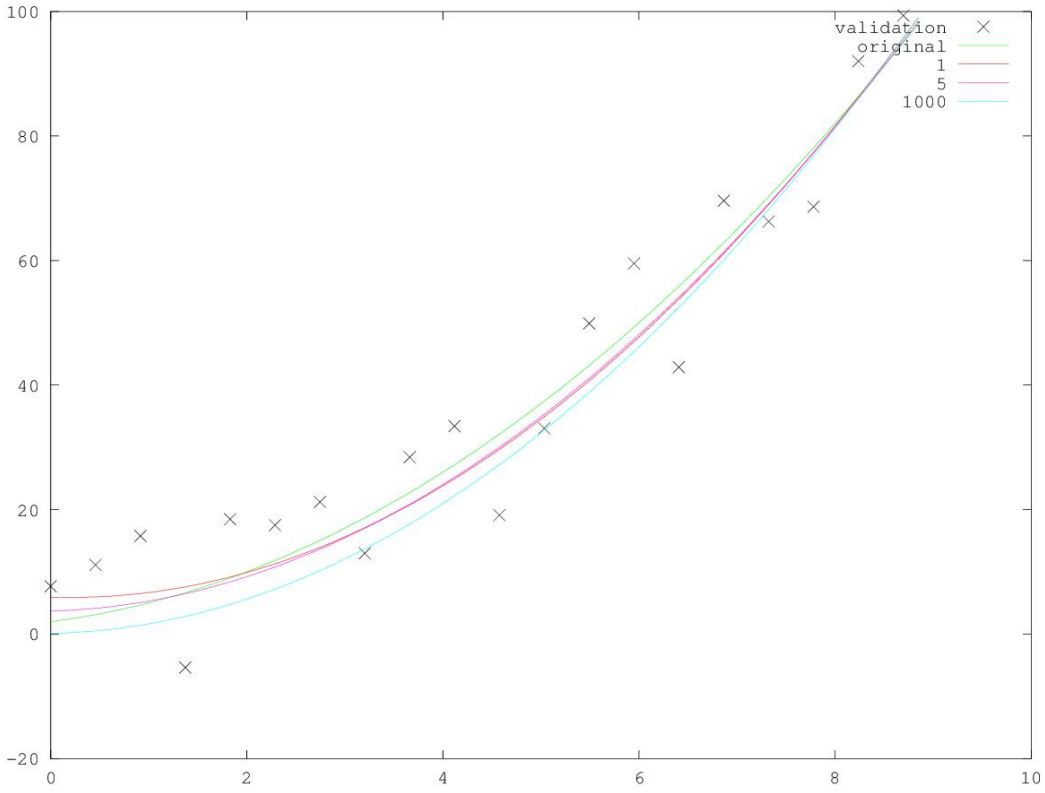

Figure 7: Plot for $f(x) = x^2 + 2x + 2$