Notes for 2017-05-03

1 Variable Projection

The case of a linear model with a non-quadratic loss function leads to one special class of nonlinear least squares. Another common special case is when a model depends linearly on some parameters and nonlinearly on others. For these problems, *variable projection* is the general strategy of eliminating the variables on which a least squares problem depends linearly.

As an example, consider the problem

$$\text{minimize } \phi(x, y) = \frac{1}{2}\|A(y)x - b\|^2$$

where $A : \mathbb{R}^p \to \mathbb{R}^{m\times n}$ depends (possibly nonlinearly) on y, but the x variables only enter the problem linearly. The *variable projection* approach involves eliminating the x variables from the equation:

$$\text{minimize } \phi(x(y), y) = \frac{1}{2}\|r(y)\|^2, \quad r(y) = A(y)x(y) - b, \quad x(y) = A(y)^\dagger b.$$

The variation of $\|r\|^2/2$ is $r^T \delta r$ where

$$\delta r = A(\delta x) + (\delta A)x;$$

and because $A^T r = 0$ (normal equations), we have

$$r^T \delta r = r^T (\delta A) x.$$

One may undertake second derivatives as an exercise in algebraic fortitude; alternately, we may apply BFGS or similar methods. Either way, we are now left with a smaller optimization problem involving the y variables alone.

2 Approximate iteration

At this point, we have considered several different methods for solving optimization problems arising from regression. The methods involved using different special structures (e.g. nonlinear least squares structure) and various simplifying approximations (e.g. Gauss-Newton and IRLS rather than

Newton). But in each case, we assumed that we were willing to look over the entire data set on which we were performing regression. But as data sets grow larger and larger, this may no longer be feasible. We conclude our discussion of regression problems with a high-level discussion of how to accelerate regression problems over large data sets. We consider three cases.

2.1 Sampling

In the worst case, we are unwilling to touch the full data set *even once*, but we are willing to compute over samples of the data. In this case, a sensible tactic might be:

1. Draw a (representative) sample from the data.
2. Do regression over the sample.
3. Repeat.

It is not always necessary to repeatedly compute over different samples, but it can be a useful way to get a sense of how accurate we believe our computed regression parameters to be. Of course, there is a lot more to sampling than this! But this is well covered in the statistics literature.

2.2 Noisy gradients

We should not sneer at simple one-shot sampling, but it has some limitations. An attractive alternative to the sample-and-regress strategy is to apply an approach like *stochastic gradient descent*, where at each step we compute an estimator of the gradient (each time based on a different sample) and step downhill in that direction. These methods are limited by two different factors: error in the gradient computation due to sampling, and the (potentially) slow rate of convergence of gradient descent. Initially, the latter effect may dominate the error, but eventually the error will always be dominated by stochastic sampling. Methods such as stochastic gradient descent are favored not because they achieve extremely accurate results quickly, but because they start making progress toward modest accuracy with relatively little effort.

Of course, there is nothing that prevents us from using small samples to make some initial progess toward a set of regression parameters, then increasing the sample size as we get closer to the solution!

3 Noisy Hessians

One-shot sampling is more a matter of understanding statistics than understanding numerical analysis. The convergence of stochastic gradient descent, while a little more numerical, ultimately again boils down to understanding variance propagation from step to step in the optimization process. On the other hand, sometimes we can get great benefit from using sampling ideas not as a replacement for the types of iterations we have discussed elsewhere in the class, but rather as an accelerator for such iterations.

To make things concrete, suppose we want to solve the linear least squares problem

$$\text{minimize } \|Ax - b\|^2$$

where $A \in \mathbb{R}^{m \times n}$. We are willing to perform the $O(mn)$ work of evaluating the residual, but n is big enough that we would prefer not to do the $O(mn^2)$ work of a factorization-based direct solve. What we can do instead in this case is:

1. Form $\tilde{A}$ by sampling $m' \ll m$ representative rows of A (though m should still be larger than n by some factor).

2. Compute the Cholesky factorization

$$\tilde{R}^T \tilde{R} = \frac{m}{m'} \tilde{A}^T \tilde{A},$$

   where we note that $\mathbb{E}[(m/m')\tilde{A}^T \tilde{A}] = A^T A$.

3. Run the fixed point iteration

$$x^{k+1} = x^k + \tilde{R}^{-1}(\tilde{R}^{-T}(A^T(b - Ax^k))).$$

   Subtracting the fixed point equation as usual gives us the error iteration

$$e^{k+1} = (I - \tilde{R}^{-1}\tilde{R}^{-T} A^T A)e^k.$$

The cost of this method is $O(m'n^2)$ for setup and then $O(mn)$ per iteration subsequently. And for well-conditioned problems, this simple sampling approach can lead to rapid convergence. Of course, we do not need to stop at fixed point iteration — we can use this type of stochastic estimate to build preconditioned Krylov subspace methods for the least squares problem as well.

If we have a *nonlinear* regression problem, the same ideas still hold, but rather than using stochastic estimators to compute $A^T A$, we compute stochastic estimates of the Hessian. To get accurate results at the end, we still want accurate gradients; but as we have seen in other circumstances, we can accelerate gradient descent quite quickly with only a rather crude approximation to a Hessian for scaling.