Computer Science Department - Rutgers University **Fall 2022**

HW 2: Optimization

16:198:501

Problem 1: Momentum Methods and Descent Directions

Consider a function

$$F(\underline{x}) = \frac{1}{2}\underline{x}^{\mathrm{T}}Q\underline{x} - \underline{x}^{\mathrm{T}}\underline{c}. \tag{1}$$

where $\underline{x}$ and $\underline{c}$ are real vectors of dimension D, and Q is a real symmetric $D \times D$ matrix.

- Show that $\nabla_{\underline{x}} F(\underline{x}) = Q\underline{x} - \underline{c}$.
- Show that if Q is positive definite, then F has a unique minimizer given by $\underline{x}^* = Q^{-1}\underline{c}$.

Traditionally, gradient descent updates take the form

$$\underline{x}_{k+1} = \underline{x}_k - \alpha_k \nabla F(\underline{x}_k). \tag{2}$$

We occasionally modify this by modifying the descent direction. **Momentum Methods** include an additional descent direction term,

$$\underline{x}_{k+1} = \underline{x}_k - \alpha_k \nabla F(\underline{x}_k) + \beta_k(\underline{x}_k - \underline{x}_{k-1}). \tag{3}$$

At step k, this moves the iterate from k to $k+1$ slightly in the direction it moved from $k-1$ to k, as though the iterate had some momentum pulling it in this direction. We want to analyze this in a little more detail in this problem.

Consider a general update of the form

$$\underline{x}_{k+1} = \underline{x}_k - \alpha_k \underline{p}_k + \beta_k \underline{q}_k, \tag{4}$$

where $\underline{p}_k = \nabla_{\underline{x}} F(\underline{x}_k)$, and $\underline{q}_k$ represents the additional modification to the descent direction we are going to make.

Show that

$$F(\underline{x}_{k+1}) = F(\underline{x}_k) + \frac{1}{2}\left(\alpha_k^2 \underline{p}_k^{\mathrm{T}} Q \underline{p}_k - 2\alpha_k\beta_k \underline{p}_k^{\mathrm{T}} Q \underline{q}_k + \beta_k^2 \underline{q}_k^{\mathrm{T}} Q \underline{q}_k\right) - \alpha_k \underline{p}_k^{\mathrm{T}} \underline{p}_k + \beta_k \underline{q}_k^{\mathrm{T}} \underline{p}_k. \tag{5}$$

- For standard Gradient Descent, taking $\beta_k = 0$, find an expression for the optimal step size α_k in terms of $\underline{p}_k$.
- For the general case, find an expression for the optimal stepsize α_k and generalized momentum factor β_k in terms of $\underline{p}_k$ and $\underline{q}_k$.

In the remainder of this problem, we want to experiment with the behavior of gradient descent and the modified momentum methods. To do so, we need a Q and a $\underline{c}$.

- Take $D = 10$.
- Write a function to generate a D-dimensional vector where each component is drawn from a standard normal distribution.
- Use this function to generate $\underline{c}$.

- Generate a $D \times D$ matrix A, where every column is a vector generated in this way.
- Take $Q = A^{\mathrm{T}} A$.

Explain why Q is almost certainly positive definite when generated this way.

Gradient Descent

- For an $\alpha > 0$ small enough to guarantee convergence, implement gradient descent for this problem. Plot the error of $||\underline{x}_k - \underline{x}^*||$, and show that it agrees with the exponential convergence we expect from the results in class. *How can you verify this?*
- Additionally, we'd like to know in what manner the iterates converge to the minimum. In going from $\underline{x}_k$ to $\underline{x}_{k+1}$, are we aimed directly at the minimizer $\underline{x}^*$, or are we off slightly? We can understand this by looking at the angle between $\underline{x}_{k+1} - \underline{x}_k$ and $\underline{x}^* - \underline{x}_k$. To get at this angle, we can plot

$$\frac{\left[\underline{x}_{k+1} - \underline{x}_k\right]^{\mathrm{T}} \left[\underline{x}^* - \underline{x}_k\right]}{||\underline{x}_{k+1} - \underline{x}_k||||\underline{x}^* - \underline{x}_k||} \tag{6}$$

  as a function of k. What does the plot suggest about how the iterates approach the minimizer?
- Are the rates of convergence of the iterates and the behavior of the approach consistent across different starting points, stepsizes, and $Q, \underline{c}$ choices?

- Instead of taking α as a constant, take α_k to be the optimal stepsize for gradient descent as found previously.
- How does this change the rate of convergence? Be as specific as you can.
- How does this change the angle of approach as the iterates converge to the minimum?
- Are these behaviors consistent?

Momentum

In this section, we include a momentum term, given by $\underline{q}_k = \underline{x}_k - \underline{x}_{k-1}$.

- For a constant $\alpha > 0, \beta > 0$, plot the error $||\underline{x}_k - \underline{x}^*||$ as a function of k to show convergence. How can you find β, α to guarantee convergence? Are these the best constants you can find?
- For the best α, β you can find in the above question, what can you say about the rate of convergence, and how does it compare to gradient descent? Can you find α, β to make the convergence rate better than vanilla gradient descent? How does it compare to optimized gradient descent?
- Again, plot the angle of approach to the minimizer for these momentum iterates. What can you say about the approach to the minimizer, and how does it compare to the previous results?
- Do the trends you observe above generalize, with $\alpha, \beta, Q, \underline{c}$? How does vanilla momentum compare with vanilla gradient descent? With optimized gradient descent?

Repeat the above, but for optimized momentum, using the optimal stepsizes α_k, β_k from before.

Is there a better direction?

We can consider the effect of the momentum term of adding a little bit of movement in a direction other than just the gradient (or rather, the negative of the gradient). This widens the space of what the iterates can explore, and in that way it makes sense that it may discover better routes to the optimum. But is this the best approach? Ideally, we'd like to move as directly towards the minimizer as possible.

What would moving directly towards the minimizer as possible 'look like', in terms of the iterates? How does this compare to the behavior of gradient descent and momentum methods as above?

An alternative approach we might take is to choose a direction $\underline{q}_k$ that is orthogonal to $\underline{p}_k$, and choose the stepsize and momentum factors to optimize motion in this orthogonal direction.

Given a vector $\underline{p}_k$, how can we generate an $\underline{q}_k$ that is orthogonal to $\underline{p}_k$?

Constructing orthogonal $\underline{q}_k$ as above, we can implement this modified momentum descent.

Implementing this modified momentum descent with optimal α_k, β_k, how does this method of choosing $\underline{q}_k$ influence the convergence rate and the directions of approach? How does it compare with momentum methods generally?

Try to come up with a better way of generating an additional direction to move in - note that you cannot use $\underline{x}^*$ or Q^{-1}, since if we knew either of these, none of this would be necessary.

Problem 2: Branch and Bound

Consider the following problem: find $x_1, x_2, \ldots, x_{10} \in \mathbb{Z}$ to maximize

$$104x_1 + 128x_2 + 135x_3 + 139x_4 + 150x_5 + 153x_6 + 162x_7 + 168x_8 + 195x_9 + 198x_{10} \tag{7}$$

such that

$$9x_1^2 + 8x_2^2 + 7x_3^2 + 7x_4^2 + 6x_5^2 + 6x_6^2 + 5x_7^2 + 2x_8^2 + x_9^2 + x_{10}^2 \leq 68644. \tag{8}$$

Note that since we want to maximize over the integers, most of our usual approaches are not going to work - no derivatives, no continuity.

However, our usual approach can still provide useful information. Because the real numbers contain the integers, the solution to the above problem over the reals must necessarily be greater than or equal to the solution over the integers. This provides a convenient upper bound on possible solutions to the problem we're actually interested in.

> What is the solution to the above problem, when the x_i are taken to be real values instead of integers? Show your work.

Additionally, note that any choice of integer values for $x_1, x_2, \ldots, x_{10}$ necessarily provides a *lower bound* on the above problem - the maximum value must be better than the result of any arbitrary choice of integers we make. If choosing integer values is done in an efficient way, such as greedily, this can generate useful lower bounds on the problem we're actually interested in. Obviously, the larger the lower bound, the better.

> Generate a greedy variable assignment to try to get a good lower bound on the value of the maximum for this problem. The larger this bound, the better.

Now to solve the original problem, we could potentially brute force it, looping over every possible feasible value for each variable. But the above results suggest a potentially better approach.

A Branch and Bound Algorithm attempts to solve this kind of optimization problem by traversing the tree of possible variable assignments, using the following observations:

- At any time, suppose we have a partial assignment of variables, and an upper and lower bound on the value the maximum attains.
- If we relax the remaining variables to be real valued, and solve for the maximum, if the result falls below the lower bound *there is no way to complete the variable assignment with integers that will surpass the lower bound.*
- In this case, we know immediately that the partial assignment must be incorrect, and we can backtrack.
- If we complete a variable assignment, and the result gives a value above the lower bound, then this variable assignment provides a *better* lower bound for the full problem, and we can utilize it moving forward.

Applying these ideas recursively, we can test assignments for some variables, determine whether they are feasible, and either explore deeper, or backtrack and test other values for variable assignment. This lets us prune the space of possible variable assignments down, and efficiently identify a maximizing assignment.

Given a partial assignment of variables, the above problem will reduce to something of the form: maximize

$$\sum_{i=1}^{n} \alpha_i z_i \tag{9}$$

subject to

$$\sum_{i=1}^{n} \beta_i z_i^2 \leq R^2 \tag{10}$$

where $z_1, z_2, \ldots, z_n \in \mathbb{Z}$.
Consider the relaxation of this to $z_i \in \mathbb{R}$, and solve for the constrained maximum in terms of $\{\alpha_i\}, \{\beta_i\}, R$.

Given a partial assignment of variables, so that the problem reduces to maximize

$$\sum_{i=1}^{n} \alpha_i z_i \tag{11}$$

subject to

$$\sum_{i=1}^{n} \beta_i z_i^2 \leq R^2 \tag{12}$$

where $z_1, z_2, \ldots, z_n \in \mathbb{Z}$, describe a way to efficiently generate an assignment of values to the variables to give a lower bound on the value of this maximum. The larger the bound the better, but not at the cost of efficiency.

Using the above solutions, implement a branch and bound algorithm to solve the original problem. What is the maximum value? How many complete variable assignments did you have to visit in order to discover the optimum?