Unit 13 – Optimization

Function Approximation

- Suppose we have a bunch of (training) data as ordered pairs \((\mathbf{x}_k, \mathbf{y}_k)\), and we would like to construct a function \(\tilde{y} = f(\tilde{x})\) that matches these ordered pairs as best as is possible

- **Example:**
  - Draw some ordered pairs in the shape of the top half of a unit circle
  - Then \(y = \sqrt{x^2 - 1}\) looks like a good approximation
  - Draw some ordered pairs on the bottom half of that circle
  - Now \(x^2 + y^2 = 1\) looks like a good approximation (but it’s not a function)

- More generally, the function need not be explicit in \(\tilde{y}\), and we just want a relationship between \(x\) and \(y\), i.e. \(f(\tilde{x}, \tilde{y}) = 0\)

- It is difficult to consider all possible functions at the same time, so we usually choose a parametric family of possible functions \(f\) (also known as a model for \(f\))

- **Example:**
  - \(f\) could be all possible circles in 2D, i.e. \((x - c_1)^2 + (y - c_2)^2 = c_3^2\) where the center of the circle \((c_1, c_2)\) and radius \(c_3\) are parameters determined to best fit the data
  - Draw some ordered pairs and show how to best fit them with a circle

- Thus, \(f(\tilde{x}, \tilde{y}, \tilde{c}) = 0\) could be families of polynomials, or circles, or a neural network, etc.

- Determine the parameters \(\tilde{c}\) that make the function best fit the training data, i.e. that make \(f(\tilde{x}_k, \tilde{y}_k, \tilde{c})\) as close to 0 as possible for all \(k\)

- Minimize \(\|f(\tilde{x}_k, \tilde{y}_k, \tilde{c})\|\) keeping in mind that \(f\) will be used for other purposes later (so don’t overfit, etc.)
  - E.g. given an input \(\tilde{x}\), evaluate \(\tilde{y} = f(\tilde{x}, \tilde{c})\), or solve the implicit \(f(\tilde{x}, \tilde{y}, \tilde{c}) = 0\) for \(\tilde{y}\)
  - E.g., given two inputs \(\tilde{x}\) and \(\tilde{y}\), use \(f(\tilde{x}, \tilde{y}, \tilde{c})\) to determine if they are wellfit by the model function \(f\)

Choosing a Norm

- The function \(f(\tilde{x}, \tilde{y}, \tilde{c})\) may have a scalar output 0 or a vector output \(\tilde{0}\) of some dimension, and in the latter case one needs to choose a norm for \(\|f(\tilde{x}_k, \tilde{y}_k, \tilde{c})\|\), e.g. \(L^1, L^2, L^\infty\), or “soft” \(L^1\) etc.

- For example: \(\|f(\tilde{x}_k, \tilde{y}_k, \tilde{c})\|_2 = \sqrt{\sum_k f(\tilde{x}_k, \tilde{y}_k, \tilde{c})^T f(\tilde{x}_k, \tilde{y}_k, \tilde{c})}\)

- There is an \(f(\tilde{x}_k, \tilde{y}_k, \tilde{c})\) for each ordered pair \((\tilde{x}_k, \tilde{y}_k)\), so one has to similarly choose a norm to combine all of these into a single expression as well

- For example, \(\sqrt{\sum_k f(\tilde{x}_k, \tilde{y}_k, \tilde{c})^T f(\tilde{x}_k, \tilde{y}_k, \tilde{c})}\) or \(\sum_k f(\tilde{x}_k, \tilde{y}_k, \tilde{c})^T f(\tilde{x}_k, \tilde{y}_k, \tilde{c})\)

- Thus, we would like to find a set of parameters \(\tilde{c}\) that minimize \(\sum_k f(\tilde{x}_k, \tilde{y}_k, \tilde{c})^T f(\tilde{x}_k, \tilde{y}_k, \tilde{c})\)

- Since all the \((\tilde{x}_k, \tilde{y}_k)\) are known, the cost function is only a function of \(\tilde{c}\), i.e. we want to minimize \(F(\tilde{c}) = \sum_k f(\tilde{x}_k, \tilde{y}_k, \tilde{c})^T f(\tilde{x}_k, \tilde{y}_k, \tilde{c})\)
Optimization
• Standard optimization problems address minimizing a cost function \( F(\vec{c}) \) possibly subject to some side constraints
• These side conditions are equations and can also contain inequalities, such as \( \vec{c}_i > 0 \) for some or all \( i \)’s
• Often these side constraints, when they exist, can be folded into the cost function if one is willing to accept the consequences (more on this later)
• When there are constraints, it is called constrained optimization, and otherwise it is called unconstrained optimization
• Since maximizing \( F(\vec{c}) \) is equivalent to minimizing \(-F(\vec{c})\) optimization is always classically approached as a minimization problem

Conditioning
• Recall that *minimizing* the residual \( r=b-Ax \) led to the normal equations \( A^TAx = A^Tb \) which squared the condition number of the original matrix \( A \)
  o This is an issue for minimization problems in general
• Solving \( F(\vec{c}) = 0 \) is generally speaking twice as easy as minimizing \( F(\vec{c}) \), since minimizing it considers the critical points where \( \frac{\partial F}{\partial c_i}(\vec{c}) = 0 \) for all \( i \)
• Having all the partial derivatives approach zero near a local minimum makes \( F(\vec{c}) \) locally flat in that region, and thus algorithms like steepest decent will struggle to compute robust downhill search directions
• One should generally expect the condition number of minimizing \( F(\vec{c}) \) to be the square of that for solving \( F(\vec{c}) = 0 \)
  o So one can only expect half as many significant digits of accuracy
  o So an error tolerance of \( \epsilon \) for \( F(\vec{c}) = 0 \) should look more like \( \sqrt{\epsilon} \) for minimizing \( F(\vec{c}) \)

Nonlinear Systems
• The critical points of \( F(\vec{c}) \) are the points where \( \frac{\partial F}{\partial c_i}(\vec{c}) = 0 \) simultaneously for all \( i \)
• Each \( \frac{\partial F}{\partial c_i} \) may readily still be its own function of all the entries of \( \vec{c} \), and we denote \( \frac{\partial F}{\partial c_i}(\vec{c}) \) as \( F_i(\vec{c}) \)
• If there are \( n \) parameters \( c_i \), then there are \( n \) potentially nonlinear equations of the form \( F_i(\vec{c}) \) that can be stacked into a vector valued function \( \vec{F}(\vec{c}) \) where the solutions of \( \vec{F}(\vec{c}) = \vec{0} \) are the critical points of \( F(\vec{c}) \)
• Generally speaking there may be no solution, any finite number of solutions, or infinite solutions