Curtis Bishop
Ben Shaw
Dave Storey

Project Proposal - CS 6830

Introduction

For our project, we propose a novel implementation of manifold-constrained regression. That is, we propose a novel implementation of an existing method of constraining the parameters of a model for regression tasks.

In 1-dimensional linear regression, one wishes to apply the model $y=mx+b$ to a dataset containing points x_i and target variables y_i. The “best line of fit” is found by optimizing the parameters m and b. Although a closed-form solution exists, the best parameters are often merely approximated due to the potentially high computational cost of finding the exact solution. Either way, the process of finding the optimal parameters m and b is an unconstrained optimization problem, since there are no apparent constraints of the parameters.

The function (of the parameters) that is optimized is known as the loss function. The loss function measures the closeness of the parameters to the optimal parameters: optimal parameters are found when the loss function is minimized.

One simple method for unconstrained optimization is gradient descent. In this method, a point is updated as follows, given a real-valued function f: x1 = x0 - eta*grad(f(p)), where x0 is the initial point, grad(f(p)) is the gradient of the function f at the point p, eta is a hyperparameter known as the learning rate, and x1 is the updated point. The iterative process of gradient descent is repeated until desired, which is typically when the change in the loss function values appears to halt.

In principle, constraints can be introduced on the parameters–the variables of the loss function. Examples include $m^2+b^2=1$, $m>1$, and $m-b^2 > 0$. The first example is an equality constraint, whereas the second two examples are inequality constraints. If a constraint is introduced, the optimization of the parameters subject to the given constraint(s) is called a constrained optimization problem. In constrained optimization, one seeks the “best” model parameters–defined by the loss function–which satisfy the given constraints.

One known method of handling constrained optimization problems is known as manifold optimization. In this technique, the constraints are thought of as defining a surface (or region) in the ambient parameter space: if the optimization parameters are forced to update only on the surface itself, the parameters will always satisfy the given constraints. Moreover, if one is able to define operations which move a point on the surface to another point on the surface, the constrained optimization has the interpretation of an unconstrained optimization problem on the surface. For our linear model, for example, the constraint $m^2 + b^2=1$ defines the surface of a

(1-dimensional) sphere: with this constraint, the problem can be recast as an unconstrained optimization problem on the 1-dimensional sphere.

Once the constrained optimization problem is cast as an unconstrained optimization problem on the surface (or region), a technique such as gradient descent can be applied: however, caution must be used. In general, the components of the gradient of a function are not merely the partial derivatives with respect to the variables of the function: although this is the case in Euclidean space in Cartesian coordinates, this is not the case in general. What is needed is called the Riemannian gradient, which involves not only the partial derivatives, but also the use of an object called the Riemannian metric. Additionally, the motion from one point to another must be along a curve known as a Geodesic curve, which is the generalization of straight lines on manifolds. Thus, the update of a point in the parameter space is a more complicated operation than in the case with Euclidean space in Cartesian coordinates.

The existing implementations of manifold optimization are extrinsic: that is, they rely on operations in the ambient space. Take, for example, the *m^2+b^1=1* constraint, which places the model parameters on the unit circle: with this constraint, one can calculate the gradient in the ambient 2-dimensional Euclidean space, then define an operation which "projects" the gradient onto the surface of the sphere.

Another approach to manifold optimization is intrinsic. For the *m^2+b^2=1* constraint, one can introduce local coordinates on the manifold–the 1-dimensional surface–from which the Riemannian metric is explicitly computed, the Geodesic curves given (or approximated), and the operations take place strictly in the 1-dimensional space. For example, one can define the coordinate *t* such that *m=cos(t)* and *b=sin(t)*: the constraint is now trivially satisfied. These equations allow one to explicitly calculate the Riemannian metric and give the equations which define the Geodesic curves, allowing one to optimize on a strictly 1-dimensional surface.

We propose implementing the intrinsic approach for a few simple examples where the regression parameters are constrained. The only known implementation of the intrinsic approach will be presented in the forthcoming publication "An Intrinsic Approach to Equality- and Inequality-Constrained Optimization on Riemannian Manifolds" (Ben Shaw, 2023[1]): however, this implementation relies heavily on symbolic software and numeric approximations for the Geodesic curves. In this project, we propose implementing the intrinsic approach (for a few simple examples) in Python without the use of symbolic software. Ideally, exact Geodesic curves will also be used.

The domain of application of constrained regression includes models in business and economics. As part of our project, we hope to better understand these domains of application and include our findings in our final report(s).

---

[1] The forthcoming publication will be available in the following Special Issue: "2nd SIAM Northern States Section Meeting, April 15-16, 2023" on Electronic Research Archive (https://www.aimspress.com/journal/era).

Preliminary Results

Thus far, we have defined the operations for the following constrained regression problem: $y=mx+b,\ m>0$. We have also begun the process of selecting a dataset for which constrained regression is suitable: we have started exploratory data analysis of a dataset which contains avocado prices and the number of sales.

Individual Responsibilities and Schedule

Below is a task list for the project. In parenthesis, we indicate the assigned group member and the anticipated date of completion.

1. Calculate/define operations for y=mx+b, m>0 in Python. (Ben, 04/10)
2. Calculate/define operations for y=ax^2+bx+c, a^2+b^2+c^2=4 in Python: proof of concept only. (Ben, 04/14)
3. Calculate/define operations for another regression problem in Python? Time permitting and depending on task 5. (Ben, 04/18)
4. Implement Stochastic gradient descent in Python using a predetermined update rule: this update rule could come from tasks 1, 2, or 3, or any constrained problem. Any number of data features, any number of model parameters. (Dave, 04/14)
5. Investigate the application of constrained regression, specifically in the context of economics/mixed market models. (Curtis, 04/17)
6. Identify (a) simple dataset(s) which fall under our domain of application: price vs. sales? Clean the data and perform exploratory data analysis. (Curtis, 04/14)
7. Implement the intrinsic approach on our dataset, including tuning the learning rate, for each constrained problem (1-3). Done after completion of 4 and 6. (Dave, 04/17–one model)
8. Prepare the group presentation. (All, 04/18)

Potential Problems

One potential problem is the ability to define the intrinsic operations in the first place. Another potential problem is the ability to utilize geodesic curves: they are difficult to compute exactly. Another potential problem in implementation is that, at least in our preliminary implementation, the hyperparameter tuning of the learning rate may be more difficult than with ordinary gradient descent.