Chapter 13

FFX: FAST, SCALABLE, DETERMINISTIC SYMBOLIC REGRESSION TECHNOLOGY

Trent McConaghy

1 Solido Design Automation Inc., Canada

Abstract

Symbolic regression is a common application for genetic programming (GP). This paper presents a new non-evolutionary technique for symbolic regression that, compared to competent GP approaches on real-world problems, is orders of magnitude faster (taking just seconds), returns simpler models, has comparable or better prediction on unseen data, and converges reliably and deterministically. I dub the approach FFX, for Fast Function Extraction. FFX uses a recently-developed machine learning technique, pathwise regularized learning, to rapidly prune a huge set of candidate basis functions down to compact models. FFX is verified on a broad set of real-world problems having 13 to 1468 input variables, outperforming GP as well as several state-of-the-art regression techniques.

Keywords: technology, symbolic regression, genetic programming, pathwise, regularization, real-world problems, machine learning, lasso, ridge regression, elastic net, integrated circuits

1. Introduction

Consider when we type “A/B” into a math package. This is a least-squares (LS) linear regression problem. The software simply returns an answer. We do not need to consider the intricacies of the theory, algorithms, and implementations of LS regression because others have already done it. LS regression is fast, scalable, and deterministic. It just works.

This gets to the concept of “technology” as used by Boyd: “We can say that solving least-squares problems is a (mature) technology, that can be reliably used by many people who do not know, and do not need to know, the details” (Boyd and Vandenberghe, 2004). Boyd cites LS and linear programming as representative examples, and convex optimization getting close. Other exam-
ples might include linear algebra, classical statistics, Monte Carlo methods, software compilers, SAT solvers\textsuperscript{1}, and CLP solvers\textsuperscript{2}.

(McConaghy et al., 2010) asked: “What does it take to make genetic programming (GP) a technology? . . . to be adopted into broader use beyond that of expert practitioners? . . . so that it becomes another standard, off-the-shelf method in the 'toolboxes' of scientists and engineers around the world?”

This paper asks what it takes to make symbolic regression (SR) a technology. SR is the automated extraction of whitebox models that map input variables to output variables. GP (Koza, 1992) is a popular approach to do SR, with successful applications to real-world problems such as industrial processing (Smits et al., 2010; Castillo et al., 2010), finance (Korns, 2010; Kim et al., 2008), robotics (Schmidt and Lipson, 2006), and integrated circuit design (McConaghy and Gielen, 2009).

Outside the GP literature, SR is rare; there are only scattered references such as (Langley et al., 1987). In contrast, the GP literature has dozens of papers on SR every year; even the previous GPTP workshops had seven papers involving SR (Riolo et al., 2010). In a sense, the home field of SR is GP. This means, of course, that when authors aim at SR, they start with GP, and look to modify GP to improve speed, scalability, reliability, interpretability, etc. The improvements are typically 2x to 10x, but fall short of performance that would make SR a “technology” the way LS or linear programming is.

We are aiming for SR as a technology. What if we did not constrain ourselves to using GP? To GP researchers, this may seem heretical at first glance. But if the aim is truly to improve SR, then this should pose no issue. And in fact, we argue that the GP literature is still an appropriate home for such work, because (a) GP authors doing SR deeply care about SR problems, and (b) as already mentioned, GP is where all the SR publications are. Of course, we can draw inspiration from GP literature, but also many other potentially-useful fields.

This paper presents a new technique for SR, called FFX – Fast Function Extraction. Because of its speed, scalability, and deterministic behavior, FFX has behavior approaching that of a technology. FFX’s steps are:

- Enumerate to generate a massive set of linear and nonlinear basis functions.
- Use pathwise regularized learning to find coefficient values for the basis functions in mapping to $y$. Pathwise learning actually returns a set of coefficient vectors; with each successive vector explaining the training data better but with greater risk of overfitting. This has the computational cost of a single LS regression, thanks to recent developments in machine learning (Friedman et al., 2010; Zou and Hastie, 2005).

\textsuperscript{1}for boolean satisfiability problems.
\textsuperscript{2}for constraint logic programming.