Genetic Programming and Evolvable Machines, 5, 259–269, 2004
© 2004 Kluwer Academic Publishers. Manufactured in The Netherlands.

Scaled Symbolic Regression

MAARTEN KEIJZER mkeijzer@cs.vu.nl
Zevenwouden 186, 3524CX Utrecht, The Netherlands

Submitted June 19, 2003; Revised October 5, 2003

Abstract. Performing a linear regression on the outputs of arbitrary symbolic expressions has empirically been found to provide great benefits. Here some basic theoretical results of linear regression are reviewed on their applicability for use in symbolic regression. It will be proven that the use of a scaled error measure, in which the error is calculated after scaling, is expected to perform better than its unscaled counterpart on all possible symbolic regression problems. As the method (i) does not introduce additional parameters to a symbolic regression run, (ii) is guaranteed to improve results on most symbolic regression problems (and is not worse on any other problem), and (iii) has a well-defined upper bound on the error, scaled squared error is an ideal candidate to become the standard error measure for practical applications of symbolic regression.

Keywords: genetic programming, linear regression, symbolic regression

1. Introduction

In Keijzer [2], the use of interval arithmetic as a replacement for protected operators and the use of linear scaling to improve the results of symbolic regression were introduced and empirically tested. The current work will take a more theoretical look at the properties of simple linear scaling and its applicability to genetic programming. Linear scaling (linear regression) is by no means new to genetic programming. Many approaches use various forms of multiple linear regression to find coefficients either inside trees [1] or to combine multiple trees [3]. Despite these approaches using multiple or generalized linear regression, the baseline method of performing symbolic regression using genetic programming is still the use of a simple error measure to determine the fitness, thus using no form of scaling at all. Here it will be proven that this choice is suboptimal, and can be replaced without the need for extra parameters with a simple linear regression on the predictions of arbitrary mathematical expressions. In [2], the effect of this simple change was empirically shown to be dramatic.

The standard method of determining the fitness (or goodness-of-fit) for the outputs of an expression $y = f(x)$ trying to fit some set of target outputs t, in genetic programming is the mean squared error:

$$E(y, t) = 1/n \sum_{i}^{n} (t_i - y_i)^2 \tag{1}$$

As an alternative, it will be shown that it is better to use the *scaled* mean squared error,

$$E_s(y,t) = E(a+by,t) = 1/n \sum_i^n (t_i - (a+by_i))^2 \tag{2}$$

Where the linear coefficients a and b (respectively called the intercept and the slope) can readily be found through a fast deterministic calculation. This letter will describe and prove basic properties of this single linear regression for a genetic programming audience. Most of the results presented below can be found in a good statistics textbook. For tutorial purposes, they are collected, proven using elementary algebra, and the importance of the results are discussed for symbolic regression.

Section 3 will prove a 'Free Lunch' theorem for E_s on search spaces of symbolic expression. It will be proven that averaged over all possible symbolic regression problems, a search procedure that employs E_s is expected to perform better than a procedure that uses E. It will also be proven that for most problems of practical interest, E_s outperforms E. These are novel results.

2. Basic properties of E_s

The scaled error measure E_s differs from the unscaled error measure E by the induction of two coefficients a and b, respectively called the *intercept* and the *slope*. Because E_s defined in Eq. (2) is quadratic in a and b, there is only one global minimum. The values for the two coefficients can be set by finding the values for which the partial derivatives w.r.t. the error is zero. If we take the scaled sum of squares measure q to be minimized as:

$$q(a,b) = \sum [t_i - (a+by_i)]^2$$

It is possible to find the minimum of this function by taking the two partial derivatives of this function and setting them to zero. After this, a and b will be located at the values where the error gradient is zero, and the error q is minimal.

$$\frac{\partial q}{\partial a} = \sum_i^n 2\,(t_i - a - by_i) \times (-1) = 0$$

$$\frac{\partial q}{\partial b} = \sum_i^n 2\,(t_i - a - by_i) \times (-y_i) = 0$$

Solving these two equations with two unknowns, first we get for a, the value:

$$-2\sum_i^n (t_i - a - by_i)) = 0$$

$$\sum_i^n a = \sum_i^n t_i - b\sum_i^n y_i \tag{3}$$

$$a = \bar{t} - b\bar{f(x)}$$

Where $\bar{x} = 1/n \sum_i^n x$, the mean, and for b,

$$\begin{aligned} 2\sum_i^n -t_i y_i + a y_i + b y_i^2 &= 0 \\ \sum_i^n \bar{t} y_i - b\bar{y} y_i + b y_i^2 &= \sum_i^n t_i y_i \\ b \sum_i^n y_i^2 - \bar{y}^2 &= \sum_i^n (t_i y_i - \bar{t}\bar{y}) \\ b &= \frac{\sum_i^n (t_i - \bar{t})(y_i - \bar{y})}{\sum_i^n (y_i - \bar{y})^2} \\ b &= \frac{\operatorname{cov}(t, y)}{\operatorname{var}(y)} \end{aligned} \tag{4}$$

Where use was made of the property of multiplication of the mean that $\sum_i^n y_i \bar{x} = \sum_i^n \bar{y}\bar{x}$ for any x, y. When y is the output of an expression that evaluates to a constant value, the variance will be zero and the value for b will be undefined. In this particular case, b is set to zero and automatically a will evaluate to $\bar{t}$, the mean value of the targets.

The derivation of the optimal a and b leads directly to a basic result on the use of scaled error.

Theorem 1. *For arbitrary y and t,*

$$E_s(y, t) \leq E(y, t)$$

Proof: This follows directly from the construction of a and b, that are set to optimal (zero error gradient) values in E_s. If $a \neq 0$ and/or $b \neq 1$ the strict form of the inequality holds. □

The effect of using the scaled error measure E_s instead of E from the perspective of genetic programming, is to change the view the selection operator has on the worth of individual expressions. As E_s rescales the expression to the optimal slope and intercept, it will focus its search on expressions that are close in shape with the target, instead of demanding that first the scale is correct. In Figure 1, this effect is depicted. Here the target function is $0.3x \sin(6.2x)$ (the first entry in the legend), and with the usual mean squared error measure E, it is already graphically obvious in Figure 1(a), that the linear expression $0.012x - 0.04$ has a far smaller error than the nonlinear expression $1 + x \sin(6x)$. When selection has to choose between these two expressions, it will thus, with a high probability, select the linear expression for propagation. However, linear scaling reveals that the nonlinear expression is in effect a much better fit than the linear one, a fact that is only revealed after rescaling the outputs of the expression. From Figure 1(b) it is suddenly clear that the scaled nonlinear expression has a far smaller error than the scaled linear expression. The use of the scaled error measure thus makes it possible to identify such expressions and let the selection operator focus on these.

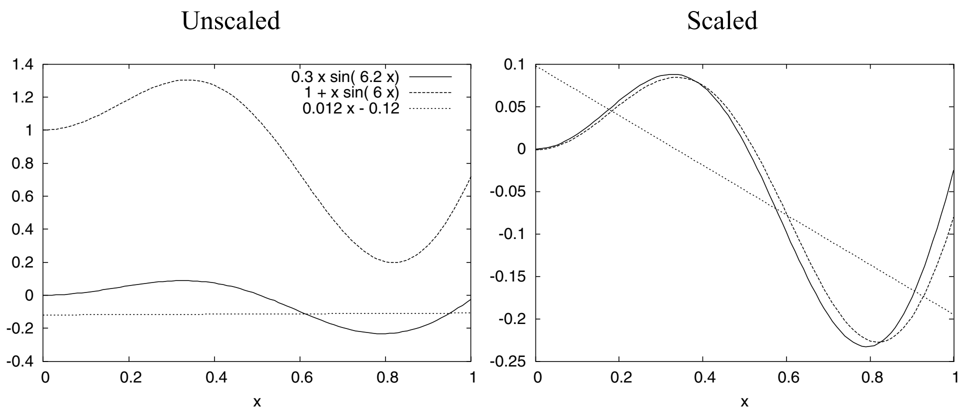

Figure 1. Selection and Scaling: the use of unscaled error measure will guide the selection operator towards a linear solution of poor quality, while the process of scaling reveals the goodness-of-fit for the nonlinear approximation.

2.1. *Upper bound on E_s*

Unlike E, whose values can become arbitrarily large for arbitrarily poor approximations, the error measure E_s has a well defined upper bound.

Theorem 2. $E_s(y, t) < \text{var}(t)$ *For arbitrary y and t.*

This is derived in the appendix.

This property is useful as it means that a standardized target variable (scaled in such a way that its mean is zero and its variance is one), will lead to a error E_s that lies between zero and one. No expression that is ever induced by genetic programming will have a value larger than this bound, and all expressions that evaluate to a constant will be mapped to the largest possible error: $1/n \sum_i^n (t_i - \bar{t})^2$. This property immediately solves the problem for symbolic regression that sometimes a run will converge on the best constant (the mean target value). As under scaled error this constant (as well as any other) has the worst possible performance value, convergence on such a value becomes exceedingly unlikely.

2.2. *Error and correlation*

Theorem 3. *The relationship between scaled mean squared error E_s and correlation r is given by*

$$E_s(y, t) = 1/n \sum_i^n (t_i - \bar{t})^2 \, [1 - r^2]$$

And thus

$$r^2 = 1 - \frac{nE_s(y,t)}{\sum_i^n (t_i - \bar{t})^2}$$

This is derived in the appendix.

Thus, the scaled error and the correlation coefficient squared—which is also known by the terms *variance explained* and *coefficient of determination*—is a simple change from minimization to maximization. From the perspective of optimization, there is no difference between minimization of scaled error E_s, or maximization of r^2 as they differ only by a factor related to the variance of the targets. This factor is constant during the run. Although in the definition of r^2, no mention is made of the slope a and intercept b, to use an expression that is induced using r^2 as a fitness measure, these coefficients still need to be calculated.

2.3. Floating point precision

Inducing arbitrarily scaled expressions on data using genetic programming often leads to problems with precision. As the techniques described in this letter attempt to rescale any expression to the same mean and variance as the target values, there is a distinct possibility that with a straightforward implementation of one of the measures above, roundoff errors will accumulate due to insufficient floating point precision. It is however possible to re-arrange the calculation in such a way that precision problems are reduced. The computation below attempts to prevent several sources of roundoff error: multiplying quantities of a different order, and loss of precision in cumulative additions by losing precision in the beginning of the addition. The recipe advocated here consists of several steps, first the targets are preprocessed and scaled such that $\sum_i^n t_i = 0$ and $\sum_i^n t_i^2 = n$. This makes sure that calculations will be done in a floating point range with good precision. Then the data is sorted on the magnitude of the targets, this to make sure that when results are added, small values will be accumulated first, before larger values will make the overall sum lose precision. When calculating the error for a new expression $y = f(x)$, the following computations are performed,

$$\begin{aligned}
m &\leftarrow 1/n \sum y_i \\
y &\leftarrow y - m \\
v &\leftarrow 1/n \sum_i^n y_i^2 \\
w &\leftarrow 1/\sqrt{v} \\
r &\leftarrow 1/n \sum_i^n (y_i \times w \times t_i)
\end{aligned}$$

The final step calculates the correlation coefficient, by using a recomputed y with a mean of zero, then scaling these to the same scale as t (by using w), and finally multiplying it with t. This recipe results in the computation of r. It is then an easy matter to either return r^2, or to provide E_s as the result of the computation. As a final cautionary measure, it is possible—and maybe advisable—to guard against very large or small values of w. Empirically, a threshold of $10^{-7} < w < 10^7$ has been found to work well.

3. Free lunch

It can be shown that regardless of the algorithm that is used, E_s is expected to outperform E on all possible symbolic regression problems. A critical assumption underlying the following proofs is that the additional computational complexity of E_s over E can effectively be ignored as a source of extra evaluations. The additional calculations necessary to calculate E_s can easily be determined and do not exceed a computation of $O(n)$. As evaluating an expression costs $O(sn)$, where s is the size of the expression, the additional computation is much smaller than a typical evaluation.[1]

The set of No Free Lunch theorems [4] has been set up to prove equivalence between algorithms using general spaces of cost functions $f \in \mathcal{F}$, some number of iterations m and any pair of algorithms a_1 and a_2 that iteratively produce expressions d. Given some performance measure Φ, one such NFL theorem reads:

Theorem 4. *For any pair of algorithms a_1 and a_2,*

$$\sum_f P(\Phi(d_m) \mid f, m, a_1) = \sum_f P(\Phi(d_m) \mid f, m, a_2)$$

In this particular case however, it is not the comparison between algorithms that is of interest, but the comparison between performance measures E_s and E. This changes the situation dramatically. The set of cost-functions now ranges over the set of possible symbolic regression problems (instances of which were denoted by t above). The quantity of interest is the probability of an algorithm reaching a certain performance level ϵ in terms of mean squared error. Two variants will be examined, namely: $\Phi(y, \epsilon) = E(y, t) \leq \epsilon$, and $\Phi_s(y, \epsilon) = E_s(y, t) \leq \epsilon$. Below, a symbolic regression problem will be described by the six-tuple $f = \langle t, x, T, F, S, D\rangle$, denoting respectively the targets, the inputs, the terminal set, the function set, the maximum size and the maximum depth.

Theorem 5. *For all algorithms a, iterations m, problems g, and error levels ϵ,*

$$P(E_s(d_m, t) \leq \epsilon \mid g, m, a) \geq P(E(d_m, t) \leq \epsilon \mid g, m, a)$$

And thus,

$$\sum_f P(E_s(d_m, t) \leq \epsilon \mid f, m, a) \geq \sum_f P(E(d_m, t) \leq \epsilon \mid f, m, a)$$

Proof: This follows directly from Theorem 1. The theorem shows that it is impossible to do worse using scaling compared to not scaling. $\square$

It is however possible to prove a stronger result:

Theorem 6. *For all algorithms a operating on a genetic programming search space, iterations m and finite error levels ϵ,*

$$\sum_f P(E_s(d_m, t) \leq \epsilon \mid f, m, a) > \sum_f P(E(d_m, t) \leq \epsilon \mid f, m, a)$$

Proof: Consider the problem $g = \langle x, x, \{x, \mathbb{R}\}, \{+\}, 3, 2\rangle$. In the error range $0 \leq \epsilon < 1/n \sum_i^n (x_i - \bar{x})^2$, the solution set for E_s, $R_\epsilon^{E_s}$ is $\{x, x + x, x + c, c + x\}$, for arbitrary constants c. This is caused by the scaling property, where all these expressions will be scaled so they will evaluate to x. For $\epsilon \geq 1/n \sum_i^n (x_i - \bar{x})^2$, the solution set $R_\epsilon^{E_s}$ is the entire search space, as also the constant expressions $\{c, c_1 + c_2\}$ will have entered the solution set, and thus $P(E_s(d, x) \leq 1/n \sum_i^n (x_i - \bar{x})^2) = 1$ (Theorem 2). For E, however, all expressions of the form $\{x + c, c + x\}$, where $c^2 > \epsilon$, are not part of the solution set R_ϵ^E. As no constant expression c enters the solution set before the upper bound $\epsilon = 1/n \sum_i^n (x_i - \bar{x})^2$ is reached, and because $x + x$ is already part of the solution set $R_0^{E_s}$, for this particular problem, $R_\epsilon^E \subset R_\epsilon^{E_s}$, for all error levels ϵ.

Now consider a random searcher a_r that samples solutions from the search space using some fixed probability function. Because $R_\epsilon^E \subset R_\epsilon^{E_s}$, $P(E_s(d, x) \leq \epsilon \mid g, 1, a_r) > P(E(d, x) \leq \epsilon \mid g, 1, a_r)$ for all ϵ. As the random searcher uses a fixed distribution, this holds for arbitrary iterations m. By Theorem 5, for all other problems at worst equality will hold, which proves the theorem for a_r, summed over all possible symbolic regression problems. As it hold for a single algorithm a_r, it will hold for all algorithms a, by applying the No Free Lunch Theorem 4. This completes the proof. □

A related result that will hold for most practical problems is,

Theorem 7. *For any problem $g = \langle t, \mathbf{x}, \{x_1, \ldots, x_n, \mathbb{R}\}, \{+\} \cup F, S, D\rangle$, for which an expression Y with $E_s(Y, t) = \zeta$ exists that has size $|Y| < S - 2$ and depth$(Y) < D$, then for any finite error level $\epsilon \geq \zeta$,*

$$P(E_s(d_m, t) \leq \epsilon \mid g, m, a) > P(E(d_m, t) \leq \epsilon \mid g, m, a)$$

Proof: By examining the basic structure of the proof in Theorem 3, it can be seen that for such an Y, the solution set $R_\zeta^E \subset R_\zeta^{E_s}$. This because $R_\zeta^{E_s}$ contains all variants of Y with any single additive constant. The size and depth assumptions on Y makes sure these expressions are part of the search space. As E does not have any method to add members to its solution set that will not be added to E_s simultaneously, due to Theorem 1, the probability of success for E_s will remain above that for E for any error level above ζ. □

This theorem is useful for practical purposes. It proves that for any problem where there exists a solution (at a particular error threshold) that does not hit the upper bounds on complexity, any searcher using scaling is more likely to find an expression with a similar error than a searcher that does not employ scaling. For proving this, it was sufficient to only examine expressions that are scaled explicitly and by showing that the scaled error measure is insensitive to this. The exact benefits for scaling are however expected to be

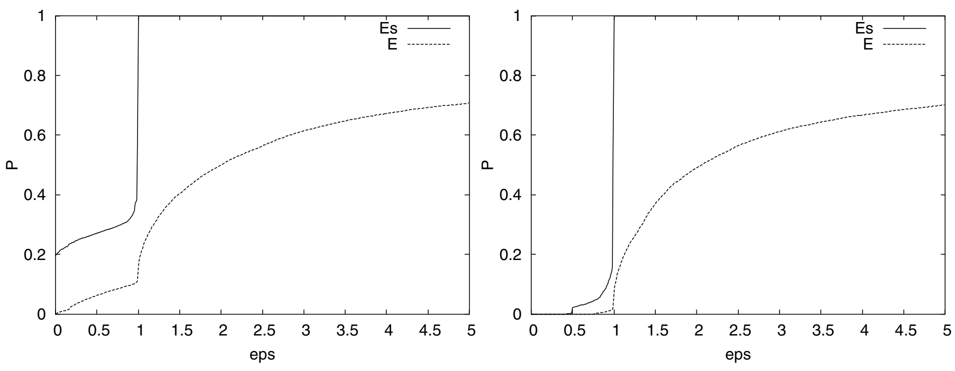

Figure 2. Plot of the probability of success with increasing success threshold ϵ (eps) under random search for the problems $t = x$ and $t = 0.3x\sin(2\pi x)$ with $T = \{x, \mathbb{R}\}$ and $F = \{+, \times, -, /\}$ for E_s and E. Only distinct expressions are used to calculate this plot.

greater than the situation that used in this proof. For instance: all expressions that, when evaluated using E, would have an error that falls above the error threshold var(t), will under E_s be mapped to a value below var(t). Such expressions can provide valuable information about the possible shape of the target function and could be used to construct the answer for the problem, while they would quite probably be discarded when selection would use the straightforward error of the expression.

An illustration of the improvements that are achievable using E_s can be found in Figure 2. Here 100,000 random programs were generated for two target functions. The probability of success for different levels of ϵ are depicted. For these problems, the effects of Theorems 6 and 7 are clear: the curve for the probability of success for E_s lies when above zero, above E. The upper bound established in Theorem 2, makes the difference especially pronounced, although also for error levels below the upper bound a significant difference can be observed. As random search is the basis for creating the initial generation of genetic programming, the figure can also be used to estimate the performance gains E_s is expected to deliver at the very beginning of a genetic programming run.

The result is not limited to the single linear regression that is studied in this paper, but directly generalizes to multiple linear regression. The result does however not generalize to a general gradient search, as the complexity of such a search is at least $O(sn)$, and thus operates in at least the same complexity class as a normal evaluation, a fact that cannot be ignored.

4. Conclusion

This letter introduces some basic properties for the use of simple linear scaling for symbolic regression. The method consists of an elementary, deterministic and cheap optimization to find the optimal slope and intercept for an expression given the training data. Unlike multiple linear regression, the method introduces no extra parameters or limitations to the genetic programming system. The work is self-contained and provides proofs for optimality of the approach, upper bounds, the relationship between linear scaling and correlation, as well as implementation issues focusing on floating point precision. These properties lead to a 'Free Lunch' theorem, where it is proven that summed over all problems, the probability of finding a solution using linear scaling is larger than the success probability using the standard mean squared error. It is also proven, that for any problem with minimal restrictions on the size and depth of solutions for that problem, using linear scaling as the error measure is expected to perform better than using unscaled mean squared error. These so-called 'Free Lunch' results are a formalization of the intuition that a cheap deterministic optimization can improve the searching capability of any search algorithm. In this particular case, the optimization is very cheap and has strong optimality guarantees.

Combining these properties and guarantees about the performance of scaled mean squared error for symbolic regression, it can be concluded that scaled mean squared error (or equivalently r^2) is an ideal candidate to become a standard performance measure in practical applications of symbolic regression for which the *squared error* needs to be minimized. Especially in comparative studies with more general constant optimization schemes, it is advisable to use scaled error as the baseline method. When comparing constant optimization using the unscaled error measure, it can not be excluded that the observed improvements of the optimization method can be mostly attributed to the (implicit) optimization of the two linear constants, albeit in a less efficient way.

Appendix

1. Proof of $E_s(y, t) < \text{var}(t)$

$$\begin{aligned}
E_s(y,t) &= 1/n \sum_i^n (t_i - (a + by_i))^2 \\
&= 1/n \sum_i^n \left(t_i^2 - 2at_i - 2by_it_i + 2aby_i + a^2 + b^2y_i^2\right) \\
&= 1/n \sum_i^n ((t_i - by_i)^2 + a^2 - 2at_i + 2aby_i) \\
&= 1/n \sum_i^n ((t_i - by_i)^2 + (\bar{t} - b\bar{y})^2 - 2(\bar{t} - b\bar{y})(t_i - by_i)) \\
&= 1/n \sum_i^n ((t_i - by_i)^2 + \bar{t}^2 - 2b\bar{y}\bar{t} + b^2\bar{y}^2 - 2\bar{t}^2 + 4by_it_i - 2b^2\bar{y}^2) \\
&= 1/n \sum_i^n ((t_i - by_i)^2 - \bar{t}^2 + 2b\bar{y}\bar{t} - 2b^2\bar{y}^2) \\
&= 1/n \sum_i^n ((t_i - by_i)^2 - (\bar{t} - b\bar{y})^2) \\
&= 1/n \sum_i^n \left(t_i^2 - \bar{t}^2 + b^2y_i^2 - b^2\bar{y}^2 + 2\bar{t}\bar{y} - 2t_iy_i\right)
\end{aligned}$$

$$
\begin{aligned}
&= 1/n \sum_i^n \left(t_i^2 - \bar{t}^2\right) + b/n \left[\sum_i^n \left(by_i^2 - 2t_i y_i + 2\bar{t}\bar{y} - b^2\bar{y}^2\right)\right] \\
&= 1/n \sum_i^n (t_i - \bar{t})^2 + b/n \left[b \sum_i^n \left(y_i^2 - \bar{y}^2\right) - 2 \sum_i^n (t_i y_i - \bar{t}\bar{y})\right] \\
&= 1/n \sum_i^n (t_i - \bar{t})^2 + b/n \left[\sum_i^n (t_i y_i - \bar{t}\bar{y}) - 2 \sum_i^n (t_i y_i - \bar{t}\bar{y})\right] \\
&= 1/n \sum_i^n (t_i - \bar{t})^2 - b/n \sum_i^n (t_i y_i - \bar{t}\bar{y}) \\
&= 1/n \sum_i^n (t_i - \bar{t})^2 - 1/n \frac{\left[\sum_i^n (t_i y_i - \bar{t}\bar{y})\right]^2}{\sum_i^n \left(y_i^2 - \bar{y}^2\right)} \\
&\leq 1/n \sum_i^n (t_i - \bar{t})^2 \\
&< 1/(n-1) \sum_i^n (t_i - \bar{t})^2 = \mathrm{var}(t)
\end{aligned}
$$

2. Proof of $E_s = 1/n \sum_i^n (t_i^2 - \bar{t}^2)[1 - r^2]$

$$
\begin{aligned}
E_s(y, t) &= 1/n \sum_i^n (t_i - \bar{t})^2 - 1/n \frac{\left[\sum_i^n (t_i y_i - \bar{t}\bar{y})\right]^2}{\sum_i^n \left(y_i^2 - \bar{y}^2\right)} \\
&= 1/n \sum_i^n (t_i - \bar{t})^2 \left[1 - \frac{\left[\sum_i^n (t_i y_i - \bar{t}\bar{y})\right]^2}{\sum_i^n (y_i - \bar{y})^2 \sum_i^n (t_i - \bar{t})^2}\right] \\
&= 1/n \sum_i^n (t_i - \bar{t})^2 \left[1 - \left[\frac{\sum_i^n (t_i y_i - \bar{t}\bar{y})}{\sqrt{\sum_i^n (y_i - \bar{y})^2 \sum_i^n (t_i - \bar{t})^2}}\right]^2\right] \\
&= 1/n \sum_i^n (t_i - \bar{t})^2 \left[1 - r^2\right]
\end{aligned}
$$

Note

1. E_s as defined in Section 2.3 costs $4n$ additions and $3n$ multiplications versus $2n$ additions and n multiplications for E. The net difference for using scaled error is thus equivalent with evaluating 4 extra nodes per expression.

References

1. H. Iba, H. de Garis, and T. Sato, "Genetic programming using a minimum description length principle," in Advances in Genetic Programming, K. E. Kinnear, Jr. (Ed.), MIT Press, 1994, Chapt. 12, pp. 265–284.

2. M. Keijzer, “Improving symbolic regression with interval arithmetic and linear scaling,” in Genetic Programming, Proceedings of EuroGP’2003, C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, and E. Costa (Eds.), vol. 2610 of LNCS, Springer-Verlag: Essex, 2003, pp. 71–83.
3. B. McKay, M. Willis, D. Searson, and G. Montague, “Non-linear continuum regression using genetic programming,” in Proceedings of the Genetic and Evolutionary Computation Conference, W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith (Eds.), Morgan Kaufmann: Orlando, Florida, USA, 1999, vol. 2, pp. 1106–1111.
4. D. H. Wolpert and W. G. Macready, “No free lunch theorems for optimization,” IEEE Transactions on Evolutionary Computation, 1997, vol. 1, no. 1, pp. 67–82.