



Genetic Algorithms Applied to Fractional Polynomials for Power Selection: Application to Diabetes Data

B. Ndabashinze¹, G. Ustundag Siray^{2,*}, L. Scrucca³

¹Cukurova University, Faculty of Arts and Sciences, Department of Statistics, 01330 Adana, Turkey

²Cukurova University, Faculty of Arts and Sciences, Department of Statistics, 01330 Adana, Turkey

³Università degli Studi di Perugia, Department of Economics, 06123 Perugia, Italy

ARTICLE INFO

Article history:


Received	18	January	2019
Revision	14	April	2019
Accepted	18	June	2019
Available online	31	August	2019

Keywords:

Fractional Polynomials
Genetic Algorithms
Function Selection Procedure
Bayesian Information Criterion

ABSTRACT

Fractional polynomials are powerful statistic tools used in multivariable building model to select relevant variables and their functional form. This selection of variables, together with their corresponding power is performed through a multivariable fractional polynomials (MFP) algorithm that uses a closed test procedure, called function selection procedure (FSP), based on the statistical significance level α . In this paper, Genetic algorithms, which are stochastic search and optimization methods based on string representation of candidate solutions and various operators such as selection, crossover and mutation; reproducing genetic processes in nature, are used as alternative to MFP algorithm to select powers in an extended set of real numbers (to be specified) by minimizing the Bayesian Information Criteria (BIC). A simulation study and an application to a real dataset are performed to compare the two algorithms in many scenarios. Both algorithms perform quite well in terms of mean square error with genetic algorithms that yield a more parsimonious model comparing to MFP Algorithm.

 2019 Turkish Journal of Forecasting by Giresun University, Forecast Research Laboratory is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/).

1. Introduction

Linear regression models are statistic techniques that are often used to examine relationships between a response variable (dependent variable) and a set of covariates (independent variables) based on some assumptions, whose one is the existence of a linear relationship between the dependent and the independent variables. However, in many cases, especially in medical research, this assumption is violated and other alternatives are suggested such as categorization (the problem of cutpoints), splines functions (the problem of number and position of knots), and higher-order polynomials (the problem of overfitting). [1] introduced fractional polynomials to model the nonlinear relation between the response and covariates.

In a multivariable model building, fractional polynomials are used to select variables and functions through an MFP algorithm that allows selecting powers in a prefixed set $S = \{-2, -1, -0.5, 0, 0.5, 1, 2, 3\}$ using multiple testing based on the Likelihood ratio test procedure, see [2]. Our proposed approach uses a stochastic search technique termed Genetic algorithms to select powers by minimizing the Bayesian information criterion in the extended search grid.

* Corresponding author.

E-mail addresses: gustundag@cu.edu.tr (Gülesen Üstündağ Şiray)

$P = \{NA, -3, -2.5, -2, -1.5, -1, -0.5, -0.25, 0, 0.25, 0.5, 1, 1.5, 2, 2.5, 3\}$, where NA represents the non-selection of a variable.

Throughout this paper, we will refer to this approach as the GAFP algorithm. Simulations based on an artificial dataset from Breiman [3] and Friedman [4] are performed taking into account many scenarios depending on sample size and error variance to compare both MFP and GAFP algorithms.

We apply both algorithms to a diabetes dataset used in [5], and compute different statistic measures to compare the goodness of fit and predictive performance of the two algorithms. Results reveal that the two algorithms are powerful to select relevant variables and their functional form. They both provide parsimonious models that are interpretable and generalizable.

2. Fractional Polynomials and MFP Algorithm

2.1. Definition and shape

Fractional polynomials are a generalization of conventional polynomials that include negative and fractional powers. More formally, for a univariate case, a fractional polynomial transformation of order (degree) $m \geq 1$ is defined as:

$$FP_m(x; \beta; p_j) = \sum_{j=0}^m \beta_j H_j(x) \tag{1}$$

where β_1, \dots, β_m are regression coefficients, $p_0 = 0$ and $H_j(x)$ a particular type of power function recursively defined as

$$H_j(x) = \begin{cases} x^{p_j} & p_j \neq p_{j-1} \\ H_{j-1}(x) \ln(x) & p_j = p_{j-1} \end{cases} \tag{2}$$

with $j = 1, 2, \dots, m$ and $H_0(x) = 1$.

Remarks:

- In the previous definition, powers p_j belongs to a predefined set $S = \{-2, -1, -0.5, 0, 0.5, 1, 2, 3\}$, with x^0 denoting $\ln(x)$ as introduced by [1].
- The term $FP_m(x; \beta; p_j) = \sum_{j=0}^m \beta_j H_j(x)$ is a linear predictor, therefore fractional polynomials can be applied to all models that use linear predictors such as logistic regression models, Cox models, etc.

Examples:

- A fractional polynomial of first degree with power $p = 0.5$ is given by the transformation: $FP_1(x) = x^{0.5}$.
- A fractional polynomial of the second degree with different powers $p(-2,0.5)$ is given by the transformation: $FP_2(x) = x^{-2} + x^{0.5}$.
- A fractional polynomial of the second degree with repeated powers $p(2,2)$ is given by the transformation: $FP_2(x) = x^2 + x^2 \ln(x)$.

The univariate FP definition described previously can be extended to FP of multiple continuous covariates and is called a multivariable FP (MFP) model. With K continuous covariates x_1, \dots, x_k ; the linear predictor is given by:

$$FP_m(x; p_1, \dots, p_k) = \beta_0 + \sum_{k=1}^K \sum_{j=1}^m \beta_{kj} H_j(x_k, p^{(k)}) \tag{3}$$

where $H_j(x_k, p^{(k)})$ is defined as in (2), k indexes and m_k the degree of a fractional polynomial correspondent to the variable x_k , β_0 the global intercept and β_{kj} the coefficient for the k^{th} FP_m model and the j^{th} transformation.

[6] suggested to consider FP_1 and FP_2 families since models with a degree higher than two are not often required in multivariable analysis. Compared to conventional polynomials of the same degree, Fractional polynomials provide many potential improvements when fitting.

2.2. Estimation and MFP algorithm

Regression coefficients in fractional polynomial models are estimated conditionally on powers in a set S . Once powers in S are selected, these coefficients are estimated using the maximum likelihood method. Fractional polynomials models having the same degree are compared using the deviance, i.e. minus twice the maximized loglikelihood of the model. The best-fitting model is the one with small deviance value. Fractional polynomials

models having a different degree are compared using the deviance difference (Likelihood ratio test) on some degree of freedom.

For practical use, [6] stated that a fractional polynomial of degree m has approximately $2m$ degree of freedom. Hence an FP_2 has 4 df; 2 df for the two powers and other 2 df for the two regression coefficients. A linear model has 1 df; an FP_1 has 2 df; one for the unique power and another for the one regression coefficient.

For the univariate case, [6] used the function selection procedure (FSP) to select the best univariate fractional polynomial model. This function selection procedure is called RA_2 in [7] and in [8]. In the following description of FSP, the linear function will be preferred unless the data require a more complex function. Before implementing the FSP, the user must first choose the nominal p -value and the degree m of the most complex function to be used. Recall that the FSP preserves the overall type I error probability at a chosen level α . We explain the FSP steps according to [6]. It runs as follows:

- The first step is of the overall relationship of the outcome and the predictor x . FP_2 model is tested with the null model using 4 df. If the test is not significant, we conclude that the predictor x doesn't affect the outcome, otherwise we continue.
- The second step checks the evidence for non-linearity. FP_2 is tested with a linear model using 3 df. If the test is not significant, we conclude that the model is linear (straight line), otherwise, we continue.
- The last step checks the complexity of the model. FP_2 model is tested with FP_1 model using 2 df. If the test is not significant, FP_1 is chosen as the final model, otherwise, the final model is FP_2 model.

In R statistical software, the default choices of significance level α and the degree m are 0.05 and 2 respectively. Recall that the user must also choose two different significance levels α_1 for variable selection (used in step 1) and α_2 for function selection (used in steps 2 and 3).

For a multivariable case, the FSP is used as a building block from which multivariable model building is constructed.

- Suppose we have K variables in the model, before applying the MFP algorithm, significance levels α_1 for variable selection, α_2 for function selection must be chosen included the maximum degree for each variable m_1, \dots, m_k . Defaults are $\alpha_1 = \alpha_2 = 0.05$ and $m_1 = m_2 = \dots = m = 2$.
- The full linear model is fit and the variables are ranked according to their Wald's test statistic (p-value) from the most significant to the least.
- The first cycle begins by applying the FSP defined previously to the most significant variable in the list and its functional form is obtained. If the variable is categorical or binary, the joint significance of its dummy variable(s) is tested at the α_1 level. If the test is significant, the binary variable is retained, otherwise, it is dropped. During the first cycle, FSP is applied to each continuous variable until the least significant. All remaining variables are included in the model as adjustment terms every time when FSP is applied to a variable. The first cycle ends when all variables are revisited.
- The second cycle begins with all variables and functional forms selected in the first cycle. The cycle begins by applying the FSP to the most significant variable and all remaining variables stay in the model with their functional form found in the first cycle as adjustment terms. The procedure repeats until all variables are revisited including also variables that were not selected from the first cycle.
- The algorithm ends when the convergence condition is met. i.e. variables and functional forms for two successive cycles haven't changed.

Remarks:

- Generally, MFP needs two, three or occasionally four cycles for convergence.
- FP models are strongly affected by the covariates' outliers or covariates distribution, especially heavy-tailed covariate distribution. To dampen these effects, MFP algorithm performs some preliminary shifting (when $x \leq 0$) to ensure the positivity of the predictor and scaling to reduce the chance of numerical underflow or overflow in extreme cases, which may result in inaccuracies and difficulties when estimating the model. For shifting and scaling operations, see [6].

2.3. Example

Let's consider a banal example that consists of using the Gaussian model to predict the Chi-square Quantiles using degrees of freedom when $\alpha = 0.05$. The mathematical expression is presented as follows:

$$y(\text{Quantile}) = \chi_{\alpha=0.05}^2(x = df)$$

To get this dataset in **R** and perform the MFP algorithm (with defaults $m = 2$ and $\alpha_1 = \alpha_2 = 0.05$) on it, we use the following command:

```
> df = 1:200
> quantile = qchisq(0.05, df, lower.tail = FALSE)
> chisquare = data.frame(quantile = quantile, df = df)
> modfp = mfp(quantile~fp(df, select=0.05), family=gaussian, data=chisquare)
```

Results of the MFP algorithm is presented in Table 1.

Table 1. Chi-square data. Application of the MFP algorithm. The selected model is FP2.

Model	Deviance D	Powers	Step	Comparison	Dev. diff	p-value
FP2	0.1546073	0.5,1	1	FP2 vs Null	854076.5	<0.0001 (4 df)
FP1	478.1275	1	2	FP2 vs Linear	477.9729	<0.0001 (3 df)
Linear	478.1275	1	3	FP2 vs FP1	477.9729	<0.0001 (2 df)
Null	854076.7	-				

In this table, it is clear that the best FP_1 model has power $p = 1$ which is a linear model and that the best FP_2 model has powers $p(-0.5,1)$. Since we have only one continuous predictor, MFP algorithm applies the FSP to that predictor and the results are described in the Table as follows:

The test (based on deviance differences) of FP_2 against the null model on 4 df is significant at significance level $\alpha = 0.05$; the p-value < 0.001 is very small. We conclude that the predictor df affects the outcome quantile, hence there is an association between these two variables.

The test of FP_2 against linear model on 3 df is also significant at $\alpha = 0.05$, p-value < 0.001, implying that the association between df and quantile is non-linear.

The test of FP_2 against FP_1 model on 1 df is again significant, p-value < 0.001.

Finally, we conclude that the relationship between df and quantile is complex. FP_2 model is selected with powers $p(-0.5,1)$.

The model summary is given below

```
Call:
glm(formula = quantile ~ I((df/100)^0.5)+ I((df/100)^1),data = chisquare)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.28 -0.012 -0.002  0.01  0.04
Coefficients:
    Est SE  t-val P-val
(Intercept)  0.74 0.01  59.95  0
I((df/100)^0.5)  24 0.03  780.40  0
I((df/100)^1)  99.7 0.02 5674.49  0
(Dispersion parameter for gaussian family taken to be 0.0007848085)
Null deviance: 8.5408e+05 on 199  df
Residual deviance: 0.15461 on 197  df
AIC: -857.46
Number of Fisher Scoring iterations: 2
```

The summary shows the shifted and scaled continuous predictor df with two significant terms $I((df/100)^0.5)$ and $I((df/100)^1)$. The shift parameter was taken to be zero since predictor values are all positives. The scale parameter was taken to be 100. For this FP_2 model, the linear predictor is of the form:

$$FP_2(df) = 0.73924 + 23.91101 \times \left(\frac{df}{100}\right)^{0.5} + 99.70743 \times \left(\frac{df}{100}\right)^1.$$

3. Genetic Algorithms and Their Implementation in FPs

3.1. Implementation of GAs in FPs models

Genetic Algorithms (GAs) are stochastic search and optimization methods that work on the principle of evolution through natural selection mechanism [9]. The basic concepts of GAs were developed by [10]. The evolutionary algorithm which is based on biological evolution requires that the fitness on individual determines its ability to survive and reproduce. The evolution process of GAs is presented in Figure 1. The principle behind GAs is that they generate and maintain a population of individuals represented by chromosomes (essentially a character string similar to the chromosomes occurring in DNA). These chromosomes represent encoded solutions to a problem. First, the initial population formed by a certain number of chromosomes (encoded solutions) is generated randomly from a population.

Chromosomes then undergo a process of evolution according to the mechanism rules of selection, crossover and mutation. Chromosomes in this generation are then evaluated according to their fitness function, with the fittest surviving and the less fit being eliminated. To avoid losing good solutions, the most fitted ones, called elites, are copied directly to the next generation. Reproduction or selection selects individuals with high fitness values in the population, and through crossover and mutation of such individuals, a new population is derived in which individuals may be even better fitted to their environment.

The process of crossover involves two chromosomes changing chunks of data (genetic information) and is similar to the process of sexual reproduction. Mutation introduces mild changes into a small proportion of the population to increase its diversity. The result is a new population that evolves over time to produce better and fitter solutions to the problem at hand.

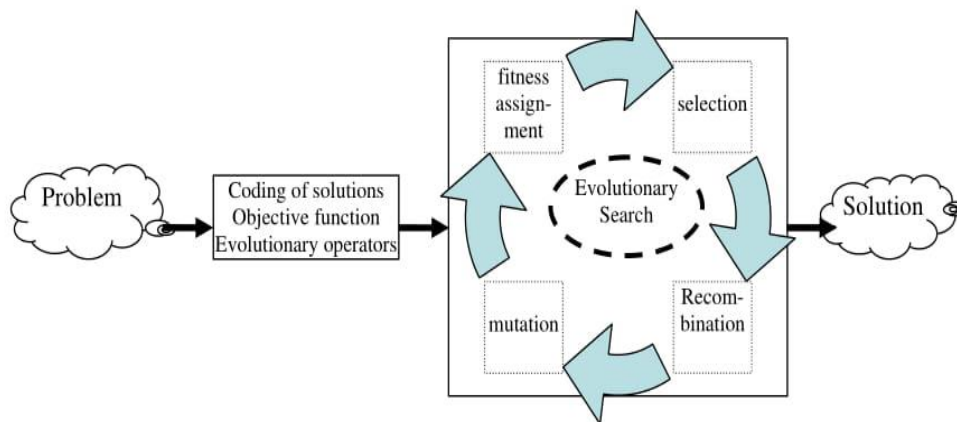


Figure 1. Evolution of GAs [11]

GAs are stochastic iterative methods and are not guaranteed to converge on an optimal solution. Therefore, search process typically ends when a pre-specified fitness value is reached, a set amount of computing time passes or until no significant improvement occurs in the population for a given number of iterations [12].

The most important genetic algorithms parameters include population size, number of generations, crossover and mutation occurrence probability, and number of elitist individuals for each generation. These tuning parameters must be set correctly since they strongly affect the results and computational time of GAs. This is the main drawback of GAs. However, their positive side is that they are parallelizable and succeed to solve an optimization problem where analytic methods fail. The details of GAs and their diverse applications can be found in [13].

3.2. Implementation of GAs in FPs models

In this section, GAs are applied to FPs to select powers in the extended set P . To do so, we use the **GA** package in the **R** statistical software [14].

3.2.1. Environment representation

To implement genetic algorithms, a search space (solution space) is needed and must be encoded in bit strings when a binary GA is used. The search space for powers is our extended set

$P = \{NA, -3, -2.5, -2, -1.5, -1, -0.5, -0.25, 0, 0.25, 0.5, 1, 1.5, 2, 2.5, 3\}$, where *NA* represents non-power selection. The set *P* has 16 powers in total, therefore it requires a 4-bit string to be represented. A one to one correspondence function **gafpEncoding** is created to link the set of powers *P* to the integer set (from 0 to 15) using the Gray code representation of those integers [15].

For variable representation, two powers are used since the most complex permitted fractional polynomial function is of second degree $m = 2$. This is the default in **mfp** package [16]. Therefore, a continuous variable will be represented by the 8-bit string. A categorical variable will be represented by 1 if it is selected in the model and by 0 otherwise.

3.2.2. Chromosome representation and initialization

In the multivariable model building, a model is a combination of variables (either continuous or categorical). Several models are possible and they are considered as chromosomes (encoded solutions) in this task. Since GAs generate a random population of chromosomes, we need to represent them the inadequate way. Suppose that our model is composed of two continuous variables with powers (NA, 3) and (1, 2); and one categorical variable. In this case, the model (chromosome) representation is given by the following $(2 \times 8) + 1 = 17$ bit string:

0000 1000 1110 1011 1.

Recall that for generating the initial population, the function **gabin_Population** in the **GA** package is used to generate a random population with specified size and length.

3.2.3. Fitness function

The main objective of GAs is to find an optimal or near-optimal solution to an optimization problem. GAs search for candidates that have a good performance which is measured in terms of the fitness function. The goal is to choose a model that has a small Bayesian Information Criterion value (BIC). This criterion is preferred over Akaike's Information Criterion (AIC) since it penalizes more the number of parameter and yields more parsimonious, interpretable and generalizable models [17].

We create the function **gafpFitness (string, X, y, family)** to be maximized, and the function **gafpPowers (string, x)** to decode the solution string provided by the function **gafpFitness (string, X, y, family)**.

3.2.4. Genetic operators

In GAs search, genetic operators are used to direct the algorithm towards an optimal solution for a problem. They are applied to a previous generation to generate a new one. For our problem at hand, we choose genetic operators that are efficient and decrease the computational time.

- **Selection:** Once an initial population is generated, candidates are evaluated according to their fitness value. The selection operator is a criterion based on fitness value. Individuals with higher fitness value have a chance to reproduce and undergo crossover and mutation. Several selection methods were proposed such as roulette Wheel Selection, tournament Selection, Rank Selection, Stochastic Selection and Elitism Selection. In this study, we use the binary tournament selection presented by [18]. The tournament selection is preferred over other fitness proportionate selection for several reasons such as lack of stochastic noise [19], parallelization and efficiency in coding [20].
- **Crossover:** The crossover operator or recombination is usually the primary operator with mutation serving only as a mechanism to introduce diversity in the population. It is applied to a pair of selected individuals with probability P_c . Parents exchange their genetic information to produce offspring. Crossover methods are: Single Point Crossover, Two Points Crossover, Multipoint Crossover, Uniform Crossover, and Arithmetic Crossover. For this study, we use uniform crossover as it doesn't exhibit positional bias caused by dividing the parent chromosome into segments for recombination, which is the case for N-point crossover operators, see [21]. Figure 2 shows an example of a uniform crossover.

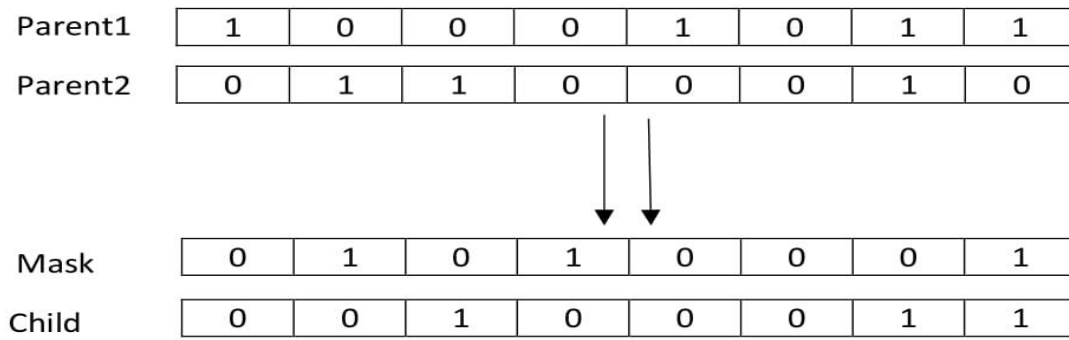


Figure 2. Uniform crossover. The chromosome with 8-bit strings

If the bit in crossover mask is 1, then the corresponding gene is copied from the first parent and if the bit in crossover mask is 0, then the corresponding gene is copied from the second parent. A new crossover mask is generated randomly for each pair of parent chromosomes. Since the quantity of crossover point is not fixed, the offspring have a mixture of genes from both parents.

- **Mutation:** Mutation introduces new genetic material into the population. It increases diversity in the population to prevent premature convergence. The mutation changes one or more randomly selected genes of a chromosome from its initial state. It happens during evolution according to a user-defined mutation probability. This probability should be small otherwise, the search will turn into a primitive random search. Figure 3 shows a mutation of an eight-bit chromosome. Two genes, the first and the fifth altered their value after mutation operation.

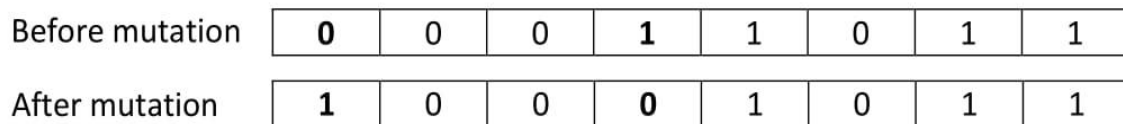


Figure 3. Mutation of a chromosome

3.2.5. Genetic algorithms parameters

As mentioned before, one of the main disadvantages of GAs is that they use several parameters for the search. These parameters can affect strongly the GAs results when badly set. Good settings of these parameters allow GAs to find better solutions in a reasonable computational time. The “No Free Lunch Theorem” states that there is no optimal parameter configuration for all problem, hence most of GA parameters depend on the problem at hand. Throughout this study, the parameters set used for the execution of our genetic algorithms are results of an empirical test and the optimal configuration is: population size (100), selection operator (tournament selection), crossover rate (0.8), crossover operator (uniform), mutation rate (0.15), mutation operator (random mutation) and maximum number of iterations (2000).

4. Simulation Studies

In this section, we perform some simulations to compare the two algorithms MFP and GAFP and examine their performance in building a multivariable model when non-linear relationships exist between the response and predictors. Weaknesses of both algorithms will be presented. To perform the MFP algorithm, we use the *mfp* function contained in the *mfp* package with current defaults: FP_2 as the most permitted complex functions, i.e. $m = 2$ or $df = 4$ and $\alpha_1 = \alpha_2 = 0.05$. As for the GAFP algorithm, the parameter settings presented above will be used.

For illustration reason, we perform simulations based on a Gaussian model using artificial data as described in [3] and [4]. The training data is generated as follows:

$$y = 10 \sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \varepsilon,$$

where the five continuous predictors come from a uniform distribution with zero mean and unit standard deviation; the error term is generated from a normal distribution with mean zero and standard deviation s . We consider different scenarios based on sample size $n = \{20,200,2000\}$ and error standard deviation $s = \{1,5,10\}$.

The test data is also generated in the same way as training. It is clear that the response variable y is related to the five predictors. Three of them x_1, x_2 and x_3 are not linearly related to the response whereas the remaining predictors x_4 and x_5 are linearly related to the response. To build a model, we use an extra variable x_6 , not related to y and having the same uniform distribution as the other five predictors in order to check in which situations both algorithms are likely to select relevant predictors with their functional form. To compare both algorithms, several statistic measures such as R-squared, BIC and the root mean square error (RMSE) for training and test data are computed. The number k of selected predictors is also given. The results are summarized in Table 2.

Simulation results for both algorithms are shown through different statistic measures. For a small sample, i.e $n = 20$, both algorithms select the wrong number of variables and are prone to overfitting. The results are good for training data and are bad for the test data. When the sample size is small, whatever the noise (error standard deviation), both algorithms perform badly and the bias selection is very large.

Table 2. Simulation results for artificial dataset.

n	s	MFP Algorithm						GAFP Algorithm					
		k	BIC	Tr R-squared	Tst R-squared	Tr RMSE	Tst RMSE	k	BIC	Tr R-squared	Tst R-squared	Tr RMSE	Tst RMSE
	1	3	99.0138	0.8088	0.502	2.0576	3.5931	6	67.3203	0.9888	0.3	0.3917	21.0833
20	5	0	133.6692	0.1605	0.0992	6.2248	7.2495	6	112.8234	0.8218	0.135	2.0485	1542.6837
	10	0	152.5553	0.1091	0.031	9.8222	11.6183	3	151.0988	0.616	0.0789	5.6506	360.5105
	1	5	829.6418	0.8988	0.8278	1.5835	2.1363	5	828.1294	0.898	0.834	1.5901	2.0936
200	5	5	1267.465	0.4696	0.3979	5.0026	5.4175	4	1262.2609	0.457	0.3808	5.0598	5.4946
	10	3	1517.9587	0.1668	0.0949	10.0226	15.7197	3	1513.6196	0.149	0.0889	10.1293	11.9796
	1	5	7849.1089	0.8881	0.8859	1.6736	1.6862	5	7856.2582	0.8885	0.8863	1.6709	1.683
2000	5	5	12351.233	0.4511	0.4398	5.1535	5.2083	5	12348.178	0.4507	0.439	5.1555	5.2119
	10	5	14987.822	0.186	0.1755	9.9929	10.1534	5	14980.73	0.1823	0.1711	10.0152	10.1807

When the medium sample size is used, i.e. $n = 200$, both algorithms select the right number of relevant predictors and perform well when noise is small, i.e. $s = 1$. However, as long as the sample becomes noisy, i.e. $s = \{5,10\}$, they select wrong predictors and fail to perform correctly. When the sample size is sufficiently large, i.e. $n = 2000$, both algorithms select exactly the true number of relevant predictors with their true functional forms and perform well even though the sample is very noisy. Values reported in Table 2 are the averaged values from the 100 simulations.

As a conclusion, both algorithms MFP and GAFP produce interpretable and transportable models when the sample has an adequate size and the noise is small. Recall that here the sample size may be small or large according to the total number of predictors used to fit the model. One cannot expect to obtain reliable and stable models when several predictors are used to model a response variable with a small sample size. For practical use, [6] suggested the sample size recommendation based on the event- per-variable (EPV) relationship. It consists of using at least 10 observations (events) per model parameter.

5. Application to Diabetes Data

This data set contains measures on 442 diabetes patients and is described in [5] who analysed it to determine the prediction model for the response variable. Ten baseline variables, *age, sex, body mass index (BMI), average blood pressure (BP)* and *six blood serum measurements (S1, S2, S3, S4, S5, S6)* were obtained for each of $n = 442$ diabetes patients, as well as the response of interest, a quantitative measure of disease progression one year after baseline. The aim was to find a parsimonious model that provides accurate predictions of response for future patients.

In this section, we carry out model selection using both algorithms MFP and GAFP on the Diabetes data and compare the results of both algorithms with the full linear model. First, we load the Diabetes data by the following commands:

```
> Diabetes = read.table("http://www.stanford.edu/~hastie/Papers/LARS/diabetes.data",
header = TRUE)
> Diabetes$SEX<-factor(Diabetes$SEX)
```

R codes and results for the three models are given below by

Full linear model

```
> modlin = glm(Y~AGE+SEX+BMI+BP+S1+S2+S3+S4+S5+S6,data=Diabetes,
  family = gaussian(link="identity"))
```

Model from MFP algorithm

```
> require(mfp)
> modfp=mfp(Y~SEX+fp(AGE,select = 0.05)+ fp(BMI,select = 0.05)+
  fp(BP,select = 0.05)+ fp(S1,select = 0.05)+
  fp(S2,select = 0.05)+ fp(S3,select = 0.05)+
  fp(S4,select = 0.05)+ fp(S5,select = 0.05)+
  fp(S6,select = 0.05), data=Diabetes,
  family= gaussian(link="identity"))
> modfp$powers
  power1 power2
BMI    1  NA
BP     1  NA
S5     1  NA
SEX2   1  NA
S1     1  NA
S2     1  NA
S4     NA NA
S6     1  2
S3     NA NA
AGE    NA NA
```

Model from GAFP algorithm

```
> require(GA)
> y = Diabetes$Y
> X = Diabetes[, -11]
> require(memoise)
> gafpFitness = memoise(gafpFitness)
> GA = ga(type = "binary", nBits = gafpNumBits(X), fitness = gafpFitness, x = X, y = y,
  family = gaussian(link="identity"), pmutation = 0.15, pcrossover = 0.8,
  popSize = 100, parallel = TRUE, maxiter = 2000, run = 100, seed = 63)
> pwrs = gafpPowers(GA@solution[1,],X)
> print(pwrs)
  power1 power2
AGE    NA NA
SEX     1 NA
BMI    NA 1
BP     1 NA
S1     NA NA
S2     NA NA
S3     NA 1
S4     NA NA
S5     NA 1
S6     NA NA
> modga = glm(Y ~ SEX + fpoly(BMI,1)+ fpoly(BP,1)+ fpoly(S3,1)+ fpoly(S5,1),
  family = gaussian(link = "identity"), data = Diabetes)
```

In the full linear model with all predictors, only four predictors *SEX*, *BMI*, *BP*, and *S5* are statistically significant at $\alpha = 0.05$. Other predictors in the model are redundant. MFP algorithm selected seven variables with eight parameters in total. The predictor *S6* has been selected with an FP_2 function which is not reasonable to have two terms for this variable. Moreover, the two terms for this predictor only have medium effect whereas other predictors selected by MFP algorithm have a strong effect on the response. GAFP algorithm which is carried out by minimizing the BIC that penalizes more parameters in the model, selected only five predictors *SEX*, *BMI*, *BP*, *S3*, and *S5*. Remark that all the five predictors have a very strong effect ($p - value < 0.05$) on the response *Y*. Different statistic measures for the three models are presented in Table 3.

Both algorithms provide models with a small number of parameters than the full linear model and perform as good as the full linear model in terms of goodness of fit and the predictive performance. GAFP algorithm seems to yield more parsimonious, interpretable, and generalizable models having approximately the same statistic measures as the model from MFP and the full linear model.

Table 3. Diabetes data. Statistic measures and selected variables for the three models

Models	Parameters in the model	BIC	R-Squared	RMSE	10-Fold Cross-validation based on RMSE
Full linear	10	4845.08	0.520	54.15	54.61472
From MFP	8	4831.62	0.520	53.58	53.91447
From GAFP	5	4816.81	0.519	54.35	54.51445

6. Summary and Future Work

In this paper, Genetic algorithms were applied to Fractional polynomials for selecting powers in an extended set in order to build multivariable models. The proposed approach, referred to as the GAFP algorithm, selects models by minimizing the BIC, whereas the MFP algorithm selects models by using multiple Likelihood Ratio Tests with some pre-defined significance levels. Both models provide reasonable results when the sample size is large enough with a less noisy dataset. Through an application to diabetes data, both models outperformed the linear model which was taken as the baseline model. GAFP algorithm seemed to provide a model that is as parsimonious, interpretable and transportable as MFP model. However, both models are shown to perform very bad when the sample is noisy and small. For practical use, a good recommendation will be to consider at least 10 observations per estimated parameter. For future work, we let for interested readers the issue of performing both algorithms by incorporating model uncertainty and shrinkage to reduce selection bias or to combine the genetic algorithms with the imputed MFP developed by [22] in order to handle covariates with missing values, often presented in datasets in which MFP models are applied. Another useful recommendation in order to reduce the computational time is to use either the Hybrid Genetic Algorithms (HGAs) [23] which incorporates efficient local search algorithms into GAs to speed up the convergence to global optimum, or GAs evolving using an Island evolution approach [23]. Here the population is partitioned in a set of sub-populations (islands) in which isolated GAs are executed on separated processor runs. Occasionally, some individuals from an island migrate to another island, thus allowing sub-populations to share genetic material.

References

- [1] P. Royston and D. G. Altman, 'Regression using fractional polynomials of continuous covariates: parsimonious parametric modelling', *Appl. Stat.*, vol. 43, no. 3, pp. 429–467, 1994.
- [2] P. Royston and W. Sauerbrei, 'Multivariable model-building: a pragmatic approach to regression analysis based on fractional polynomials for modelling continuous variables'. John Wiley & Sons, 2008.
- [3] L. Breiman, 'Bagging predictors', *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.
- [4] J. H. Friedman, 'Multivariate adaptive regression splines', *Ann. Stat.*, pp. 1–67, 1991.
- [5] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, and others, 'Least angle regression', *Ann. Stat.*, vol. 32, no. 2, pp. 407–499, 2004.
- [6] P. Royston and W. Sauerbrei, *Multivariable model-building: a pragmatic approach to regression analysis based on fractional polynomials for modelling continuous variables*. John Wiley & Sons, 2008.
- [7] G. Ambler and P. Royston, 'Fractional polynomial model selection procedures: investigation of Type {I} error rate', *J. Stat. Comput. Simul.*, vol. 69, no. 1, pp. 89–108, 2001.
- [8] W. Sauerbrei and P. Royston, 'Corrigendum: Building multivariable prognostic and diagnostic models: transformation of the predictors by using fractional polynomials', *J. R. Stat. Soc. Ser. A (Statistics Soc.)*, vol. 165, no. 2, pp. 399–400, 2002.
- [9] C. Darwin, 'The origin of species by means of natural selection, or, The preservation of favored races in the struggle for life. Vol. 1', *Int. Sci. Libr.*, 1897.
- [10] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. U Michigan Press, 1975.
- [11] S. Owais, P. Gajdoš, and V. Snášel, 'Usage of genetic algorithm for lattice drawing', *Concept Lattices Appl.*, vol. 2005, pp. 82–91, 2005.
- [12] D. E. Golberg, 'Genetic algorithms in search, optimization, and machine learning', *Addion wesley*, vol. 1989, p. 102, 1989.
- [13] R. L. Haupt and H. S. Ellen, *Practical genetic algorithms*. John Wiley & Sons, 2004.
- [14] L. Scrucca, '{GA}: A Package for Genetic Algorithms in {R}', *J. Stat. Softw.*, vol. 53, no. 4, pp. 1–37, 2013.
- [15] M. Wahde, *Biologically inspired optimization methods: an introduction*. WIT Press, 2008.
- [16] G. Ambler and A. Benner, 'mfp: Multivariable Fractional Polynomials. {R} package version 1.4. 9'. 2010.
- [17] A. A. Neath and J. E. Cavanaugh, 'The Bayesian information criterion: background, derivation, and applications', *Wiley Interdiscip.*

Rev. Comput. Stat., vol. 4, no. 2, pp. 199–203, 2012.

- [18] D. E. Goldberg and K. Deb, 'A comparative analysis of selection schemes used in genetic algorithms', *Found. Genet. algorithms*, vol. 1, pp. 69–93, 1991.
- [19] T. Blickle and L. Thiele, 'A comparison of selection schemes used in evolutionary algorithms', *Evol. Comput.*, vol. 4, no. 4, pp. 361–394, 1996.
- [20] B. L. Miller and D. E. Goldberg, 'Genetic algorithms, tournament selection, and the effects of noise', *Complex Syst.*, vol. 9, no. 3, pp. 193–212, 1995.
- [21] G. Syswerda, 'Uniform crossover in genetic algorithms', in *Proceedings of the third international conference on Genetic algorithms*, 1989, pp. 2–9.
- [22] T. P. Morris, I. R. White, J. R. Carpenter, S. J. Stanworth, and P. Royston, 'Combining fractional polynomial model building with multiple imputation', *Stat. Med.*, vol. 34, no. 25, pp. 3298–3317, 2015.
- [23] L. Scrucca, 'On Some Extensions to GA Package: Hybrid Optimisation, Parallelisation and Islands Evolution', *R J.*, vol. 9, pp. 187–206, 2017.