Model-based problem solving through symbolic regression via pareto genetic programming

Vladislavleva, E.

Publication date:
2008

Model-based Problem Solving through Symbolic Regression via Pareto Genetic Programming

Model-based Problem Solving through Symbolic Regression via Pareto Genetic Programming

Proefschrift

ter verkrijging van de graad van doctor aan de Universiteit van Tilburg, op gezag van de rector magnificus, Prof. dr. F.A. van der Duyn Schouten, in het openbaar te verdedigen ten overstaan van een door het college voor promoties aangewezen commissie in de aula van de Universiteit op vrijdag 5 september 2008 om 14.15 uur door

Ekaterina Yurievna Vladislavleva

geboren op 11 september 1978 te Moskou, Rusland.

Promotor:	prof. dr. ir. D. den Hertog
Copromotor:	dr. G.F. Smits
Overige Commissieleden:	prof. dr. W. Banzhaf
	prof. dr. A. Cuyt
	dr. M. Keijzer
	prof. dr. J.P.C. Kleijnen
	prof. dr. V.V. Toropov

To my Friends and
Soulmates

Any sufficiently advanced technology is indistinguishable from magic.

Sir Arthur Charles Clarke

Acknowledgment

This thesis would not have happened without the vision and a beautiful mind of Guido Smits, my co-supervisor from Dow Benelux B.V., who introduced me to the rich world of evolutionary computation and infected me with an urge to develop algorithms. This thesis would not have happened without the interest and dedicated coaching of Dick den Hertog, my supervisor and promotor from Tilburg University. Dick opened up the world of operations research, optimization, and meta-modeling for me. He was also the one who fearlessly hired me to do a three-year PhD when I was on the seventh month of pregnancy... Dick's confidence in me made me want to prove that he took the right decision.

Doing a PhD with support and supervision of these two bright personalities was both joy and fun. I have been very lucky with teachers my whole life, and I am especially lucky because many Teachers became Friends and Friends became Teachers. I would like to thank them all for shaping me the way I am, and for inspiring me in one way or the other to pursue this research and to complete this dissertation.

Ella Nikolaevna Yashina, who passed away far too early, made me love German at school, and was my example of a free thinker and a cosmopolitan even under the pressure of Sovjet regime. Leonid Isaakovich Zvavich gave me the best ever high-school math training and pre-destined my study at the department of mathematics at Moscow State University. Alexander Sergeevich Strogalov was my true mentor at the university. He and Sergey Gennadievich Shekhovtsov made me think about the true meaning of words and the real purpose of learning. Stef van Eijndhoven was my wanna-be-like-him director of the mathematics for industry program in Eindhoven, who convinced me that one man can win a war. Stef also found time and patience to read the first lousy versions of the introductory chapter and forced me to re-write it from scratch, which greatly improved both the style and the content of the chapter. Mark Kotanchek taught me to love Mathematica and tons of stuff on symbolic regression and life. The Kotanchek family is certainly one of the most significant discoveries I made during this PhD project.

My grandmother Nina taught me to be demanding to myself and to do any job with passion and dedication. Grandmom stays an example of a Teacher for me and I hope she will explain the beauty and simplicity of physics to my daughter

in a couple of years as she did it to me. My father taught me to love Nature and to take life easy. My wise mother taught me to take independent decisions from the early years, to reason critically, to keep joy in the heart, and to always look for positive sides in any experience. I would like to thank everybody in my big family for the greatest gift of all — for raising me in a world of love and respect, i.e. for making me a happy person. My beloved sister Olya literally made this thesis happen, because in 2006 she took a gap year and exchanged an energetic life of a young lady in Moscow, Russia, by the energy-consuming life of a sister-baby-sitter in Breda, the Netherlands. Thanks to Olya, her love and support, I could temporarily abstract myself from the joys of diapers and burps during the working hours, and focus on the joys of doing research, always knowing that my baby is fed, loved, and taken care of.

My baby daughter Sonya, who recently turned three, has been the greatest source of joy, love, hope, and strength, which most probably influenced the style in which this thesis is written. Sonya taught me many things. Her persistent lessons that sleeping is unnecessary for young mothers like me were very hard to master, but appeared to be particularly useful in the last stages of writing this dissertation.

Finishing this thesis is a big personal accomplishment for me and I would like to express my gratitude to people who were my inspiration. I thank Rick Riolo for letting me join the workshops on genetic programming in theory and practice at the University of Michigan and Una-May O'Reilly, Jaison Daida, Arthur Kordon, Trent McConaghy, Stu Card, Lee Spector, Wolfgang Banzhaf, Maarten Keijzer, Bill Worzel, Lee Jones, Bill Tozier, Terry Soule, and Ying Becker for inspiring discussions on life and GP, many of which encouraged me to keep going in the hard year of 2006.

I thank colleagues from the CentER group of Tilburg university, especially Etienne de Klerk, Annemiek Dankers, Bauke Jansen, Sandra de Bruin, Heidi Ket, and Sebastian Gryglewicz for making my time in Tilburg enjoyable. I thank Jaap den Doelder, Elsa Jordaan, Michael de Graaf, Olena Zavinska, Jerome Claraq, Sjoerd de Vries, Jolanda Kotvis, Lizzy Vinje, Gert Berendsen, Sylvie Vervoort, Tom Verbrugge, Alain Sienaert, and Guido Smits for making my work at Dow Terneuzen great fun. Special thanks go to Jaap, Olena, Guido, Sylvie and Sjoerd who helped me settle in as well as get out of Terneuzen.

I am grateful to Wolfgang Banzhaf, Maarten Keijzer, Vassili Toropov, Jack

Kleijnen, Annie Cuyt, and also to Mark Kotanchek and Arthur Kordon for finding time to review this dissertation. I especially thank Jack, Vassili, and Mark for detailed comments and valuable suggestions.

I wanted to do this PhD because it was a unique opportunity to stay on the edge between academia and industry, to understand the intriguing power of evolution by "intelligent" design, to maximize the learning experience in a minimal amount of time, and to develop the intuition of what industry wants, how academia can help it, and where (and whether) I can make a difference.

I did finish this PhD because so many great people whom I respect and admire believed in me. I hope I did meet their expectations, because I am here, and I am learning and growing.

Ekaterina Vladislavleva

June 2008

Samenvatting

De focus van dit proefschrift is de identificatie van het verband tussen bepaalde input-output gegevens door middel van symbolische regressie. De uitdagende taak van symbolische regressie is een echt of gesimuleerd systeem of een proces te identificeren en te vertalen in een expliciete functie, gebaseerd op een beperkt aantal observaties van het gedrag van dit systeem.

Het bestudeerde systeem wordt gekenmerkt door een aantal belangrijke controleparameters, die voor een waarnemer beschikbaar moeten zijn, maar die meestal moeilijk, of via een tijdrovend experiment of slechts met hoge kosten te meten zijn. Een andere mogelijkheid is dat deze parameters slechts na lange tijd of met hoge computerkosten kunnen worden gesimuleerd of waargenomen.

Door middel van empirisch modelleren wordt geprobeerd om deze kritieke controlevariabelen via andere beheersbare variabelen uit te drukken die of gemakkelijker zijn te meten, nauwkeuriger kunnen worden gemeten, goedkoper zijn om te simuleren, enz.

Symbolische regressie maakt het mogelijk dergelijke uitdrukkingen van essentiële proceskenmerken of bepaalde reactievariabelen te verkrijgen in symbolische vorm. Deze uitdrukkingen worden empirische input-output modellen genoemd en hebben als inputs de meer eenvoudig te meten controleparameters.

Voorbeelden hiervan zijn (1) structuur-activiteit verhoudingen in geneesmiddelen, waar de activiteit van een medicijn wordt bepaald door de fysieke structuur van moleculaire componenten, (2) structuur-eigenschap verhoudingen in materiaalkunde, waar de kwaliteiten van een product, zoals glans, opaciteit, geur, of stijfheid van een product wordt gerelateerd aan de samenstelling en verwerkingsvoorwaarden, of (3) economische modellen, b.v. rendement van een investering als functie van aandeelkoersen en economische parameters.

De industrile modelleringsproblemen die geschikt zijn voor symbolische regressie hebben twee belangrijke kenmerken: (1) geen of weinig informatie is bekend over het onderliggende systeem dat de gegevens produceert, en daarom kunnen geen veronderstellingen over de mogelijke modellen worden gemaakt; (2) de beschikbare gegevens zijn hoog-dimensionaal, niet gebalanceerd, en met of een overvloedig of een ontoereikend aantal steekproeven.

Om aannemelijke modellen met realistische tijd- en computerinspanningen te ontdekken, exploiteert symbolische regressie een stochastische zoekmethode,

die op kunstmatige evolutie van modeluitdrukkingen wordt gebaseerd. Deze methode, die genetische programmering wordt genoemd, zoekt geschikte uitdrukkingen voor de output-variabele in de ruimte van alle geldige formules die een minimaal aantal inputvariabelen, operatoren en constanten bevatten. De mogelijke operatoren worden vooraf vastgelegd.

Bij elke iteratie van het genetische programmeersysteem wordt een voldoende grote hoeveelheid verschillende formules gevalueerd, wordt vervolgens de deelverzameling van formules geselecteerd die het beste voldoet aan een aantal door de gebruiker bepaalde prestatiecriteria, en worden vervolgens de beste formules uit deze deelverzameling gerecombineerd om een gevarieerde reeks potentile oplossingen voor de volgende stap tot stand te brengen. Deze benadering wordt genspireerd door principes van natuurlijke selectie, waar de nakomelingen die goede eigenschappen van beide ouders erven hun kansen voor overleving, aanpassing, en verdere propagatie verhogen. De uitdaging voor evolutionaire algoritmes is het vinden van een goed evenwicht tussen de exploitatie of het verfijnen van de reeds gevonden oplossingen en de exploratie van nieuwe gebieden van de onderzoeksruimte waar nog betere oplossingen kunnen worden gevonden.

Het feit dat symbolische regressie via Pareto GP geen veronderstellingen aan de structuur van de input-output modellen oplegt, betekent dat de modelstructuur voor een groot deel door gegevens en door selectiedoelstellingen van het evolutionaire proces bepaald wordt. Enerzijds is het een voordeel en een uniek vermogen vergeleken met andere globale benaderingstechnieken, aangezien het een potentieel heeft om eenvoudigere modellen te ontwikkelen dan bijvoorbeeld door interpolatie met veeltermen. Anderzijds is het ontbreken van beperkingen op modelstructuur de grootste uitdaging voor symbolische regressie, aangezien het de onderzoeksruimte van mogelijke oplossingen, die reeds inherent groot is, enorm verhoogt.

In dit proefschrift wordt een speciale versie gebruikt van genetisch programmeren, Pareto genetische programmering. In deze versie worden meerdere doelstellingen voor optimaliteit gecombineerd. Het gebruik van Pareto genetische programmering voor symbolische regressie heeft sterke voordelen in het creren van diverse reeksen regressiemodellen (met slechts enkele significante variabelen). Op die manier worden namelijk modellen verkregen die zowel nauwkeurig voorspellen als die voldoen aan bepaalde eisen voor structuureenvoud.

Dit proefschrift breidt de Pareto genetische programmeringsmethodologie

uit door extra generieke modelselectie en generatiestrategien die (1) modellen drijven in de richting van verminderde niet-lineariteit en verhoogde generalisatie mogelijkheden, en (2) de doeltreffendheid van het zoeken naar robuuste modellen verbeteren door zachtere doelstellingen, adaptieve fitnesssevaluaties, en verbeterde leerstrategien.

Naast de nieuwe strategien voor modelontwikkeling en modelselectie, stelt dit proefschrift een nieuwe benadering voor voor de analyse, het rangschikken, en compressie van bepaalde multidimensionele input-output gegevens, met als doel de informatie-inhoud van deze gegevensreeksen meer in evenwicht te brengen.

Om bijdragen van dit onderzoek in de context van real-life probleem oplossingen te plaatsen, exploiteert dit proefschrift een generiek kader van adaptief model-gebaseerd probleem oplossen zoals gebruikt in vele industrile modelleringstoepassingen. Dit kader bestaat uit herhaalde terugkoppeling van: (Deel I) generatie, analyse en aanpassing van de data, (Deel II) modelontwikkeling, en (Deel III) probleemanalyse en -reductie.

Deel I van het proefschrift bestaat uit Hoofdstuk 2 en behandelt gegevens-analyse. Het bestudeert manieren om multi-dimensionele input-output gegevens in evenwicht te brengen voor succesvolle verdere modellering. Het hoofdstuk stelt verscheidene nieuwe methodes voor voor interpretatie en manipulatie van bepaalde hoog-dimensionale input-output gegevens, zoals het relatief wegen van gegevens, het rangschikken van gegevens in volgorde van stijgend belang, en het bepalen van de samendrukbaarheid en de informatie-inhoud van een multi-dimensionele gegevensreeks. Alle methodes exploiteren de geometrische structuur van de gegevens en de relatieve afstanden met dichtbij gelegen datarecords, en behandelen de output-waarden verschillend, rekening houdend met het feit dat de gegevens tot een bepaald multi-dimensionaal oppervlak behoren.

Deel II van het proefschrift bestaat uit Hoofdstukken 3-7 en behandelt de modelinductiemethode - Pareto genetische programmering (Pareto GP). Omdat tijd tot het bereiken van een oplossing, of nauwkeuriger, tijd-tot-overtuigende oplossing een belangrijke praktische uitdaging is van evolutionaire zoekalgoritmen en dus ook voor Pareto GP, behandelen de hoofdstukken van Deel II diverse algoritmische verbeteringen van Pareto GP. Deze leiden tot de snellere ontdekking van betere oplossingen, d.w.z. oplossingen van voldoende kwaliteit bij een kleinere cpu-inspanning, of van aanzienlijk betere kwaliteit bij dezelfde cpu-inspanning.

In Hoofdstuk 3 wordt een algemene beschrijving van de Pareto GP methodologie voorgesteld in een kader van evolutionair zoeken met een herhaalde iteratie over de stadia van modelgeneratie, modelevaluatie, en modelselectie.

In Hoofdstuk 4 wordt een nieuwe strategie voor modelselectie door expliciete niet-lineariteits controle voorgesteld. Een nieuwe complexiteitsregel gebaseerd op de mate van niet lineair zijn van symbolische modellen wordt gentroduceerd in Pareto GP en met succes gebruikt als onafhankelijk optimaliseringscriterium. Er wordt aangetoond dat dit criterium ook met een criterium gebaseerd op expressielengte kan worden afgewisseld, met als gevolg de ontwikkeling van modellen met betere extrapolatie mogelijkheden.

In Hoofdstuk 5 wordt een nieuwe manier voor het evalueren van de fitness van modellen gentroduceerd. Hierin worden de methodes gebruikt om gegevens in evenwicht te brengen zoals beschreven in Hoofdstuk 2. In deze modificatie wordt een gewijzigde definitie van modelvoorspellingsfout voor niet gebalanceerde gegevens gexploiteerd. Twee verschillende methodes om gegevens in evenwicht te brengen worden voorgesteld. De eerste methode gebruikt de gewichten die de relatieve informatie-inhoud van de input-output gegevens weergeven direct bij het uitvoeren van de regressie met de gewogen voorspellingsfout. De tweede methode reduceert de reeks van input-output gegevens tot een kleinere groep van gelijkwaardige informatie-inhoud en voert dan standaard regressie op de kleinere groep uit met verbeterde exploratie.

In Hoofdstuk 6 wordt een alternatieve strategie voor modelevaluatie in Pareto GP gentroduceerd, gebaseerd op 'goal softening' en ordinale optimalisatie. Deze strategie is ontworpen om het evenwicht tussen exploratie en exploitatie in het evolutionair zoeken te verbeteren.

In Hoofdstuk 7 wordt een nieuwe methode voor incrementele evoluties op essentile data records voorgesteld, die ideen van gebalanceerde gegevens (van Hoofdstuk 2) en 'goal softening' (van Hoofdstuk 6) in n nieuw kader combineert. In dit kader worden de data records, die in dalend belang worden gesorteerd, incrementeel toegevoegd aan het modelleringssysteem dat met een zeer kleine subgroep en een zeer grote modelpopulatie begint. De modelpopulatie vermindert in de loop van de evolutie zodanig dat het rekenbudget per iteratie constant blijft.

De prestatie verbeteringen van de voorgestelde methodes ten opzichte van standaard Pareto GP worden vergeleken voor een aantal test problemen en getest op hun statistische significantie.

Deel III van het proefschrift is gewijd aan Probleem Analyse en Reductie. Het veronderstelt dat de modellen die in Deel II worden ontwikkeld, in detail worden onderzocht en zorgvuldig worden genterpreteerd om enkele inleidende gevolgtrekkingen over de moeilijkheid van het modelleringsprobleem te maken. Deel III bestaat uit Hoofdstuk 8 en behandelt de praktische aspecten van het gebruiken van symbolische regressieresultaten voor het schatten van de moeilijkheidsgraad van het probleem, in het bijzonder voor relevante input variabele selectie, convergentie - identificatie, betrouwbaarheidsevaluatie, en adaptieve data collectie.

Summary

The main focus of this dissertation is identification of relationships from given input-output data by means of symbolic regression. The challenging task of symbolic regression is to identify and express a real or simulated system or a process, based on a limited number of observations of the system's behavior.

The system under study is being characterized by some important control parameters which need to be available for an observer, but usually are difficult to monitor, e.g. they need to be measured in a lab, simulated or observed in real time only, or at high time and computational expenses. Empirical modeling attempts to express these critical control variables via other controllable variables that are easier to monitor, can be measured more accurately or timely, are cheaper to simulate, etc. Symbolic regression provides such expressions of crucial process characteristics, or, response variables, defined (symbolically) as mathematical functions of some of the easy-to-measure input variables, and calls these expressions empirical input-response models. Examples of these are (i) structure-activity relationships in pharmaceutical research, which define the activity of a drug through the physical structure of molecules of drug components, (ii) structure-property relationships in material science, which define product qualities, such as shininess, opacity, smell, or stiffness through physical properties of composites and processing conditions, or (iii) economic models, e.g. expressing return on investment through daily closes of S&P 500 quotes and inflation rates.

Industrial modeling problems that are tractable for symbolic regression have two main characteristics: (1) No or little information is known about the underlying system producing the data, and therefore no assumptions on model structure can be made; (2) The available data is high-dimensional, and often imbalanced, with either abundant or insufficient number of samples.

To discover plausible models with realistic time and computational efforts, symbolic regression exploits a stochastic iterative search technique, based on artificial evolution of model expressions. This method, called genetic programming looks for appropriate expressions of the response variable in the space of all valid formulae containing a minimal set of input variables and a proposed set of basic operators and constants.

At each step, the genetic programming system considers a sufficiently large quantity of various formulae, selects the subset of the "best" formulae according

to certain user-defined criteria of goodness, and (re)combines the best formulae to create a rich set of potential solutions for the next step. This approach is inspired by principles of natural selection, where the offspring that inherits good features from both parents increases the chances to be successful in survival, adaptation, and further propagation. The challenge and the rationale of performing evolutionary search is to balance the exploitation of the good solutions discovered so far, with exploration of the new areas of the search space, where even better solutions may be found.

The fact that symbolic regression via genetic programming (GP) does not impose any assumptions on the structure of the input-output models means that the model structure is to a large extent determined by data and also by selection objectives used in the evolutionary search. On one hand, it is an advantage and the unique capability compared with other global approximation techniques, since it potentially allows to develop inherently simpler models than, for example, by interpolation with polynomials or spatial correlation analysis. On the other hand, the absence of constraints on model structure is the greatest challenge for symbolic regression since it vastly increases the search space of possible solutions which is already inherently large.

A special multi-objective flavor of a genetic programming search is considered, called Pareto GP. Pareto GP used for symbolic regression has strong advantages in creating diverse sets of regression models, satisfying competing criteria of model structural simplicity and model prediction accuracy.

This thesis extends the Pareto genetic programming methodology by additional generic model selection and generation strategies that (1) drive the modeling engine to creation of models of reduced non-linearity and increased generalization capabilities, and (2) improve the effectiveness of the search for robust models by goal softening, adaptive fitness evaluations, and enhanced training strategies.

In addition to the new strategies for model development and model selection, this dissertation presents a new approach for analysis, ranking, and compression of given multi-dimensional input-output data for the purpose of balancing the information content in undesigned data sets.

To present contributions of this research in the context of real-life problem solving, the dissertation exploits a generic framework of adaptive model-based problem solving used in many industrial modeling applications. This framework consists of an iterative feed-back loop over: (Part I) data generation, analysis

and adaptation, (Part II) model development, and (Part III) problem analysis and reduction.

Part I of the thesis consists of Chapter 2 and is devoted to data analysis. It studies the ways to balance multi-dimensional input-output data for making further modeling more successful. Chapter 2 proposes several novel methods for interpretation and manipulation of given high-dimensional input-output data such as relative weighting the data, ranking the data records in the order of increasing importance, and accessing the compressibility and information content of a multi-dimensional data set. All methods exploit the geometrical structure of the data and relative distances to nearest-in-the-input space neighbors. All methods treat response values differently, assuming that the data belongs to a response surface, which needs to be identified.

Part II of the thesis consist of Chapters 3-7 and addresses the model induction method - Pareto genetic programming. Since time to solution, or, more accurately, time-to-convincing-solution is a major practical challenge of evolutionary search algorithms, and Pareto GP in particular, Part II focuses on algorithmic enhancements of Pareto GP that lead it to the discovery of better solutions faster (i.e. solutions of sufficient quality at a smaller computational effort, or of considerably better quality at the same computational effort).

In Chapter 3 a general description of the Pareto GP methodology is presented in a framework of evolutionary search, as an iterative loop over the stages of model generation, model evaluation, and model selection.

In Chapter 5 a novel strategy for model selection through explicit non-linearity control is presented. A new complexity measure called the order of non-linearity of symbolic models is introduced and used successfully either as an independent optimization criterion or alternated with expressional complexity, which is both cases leads to the development of models with improved extrapolative capabilities.

In Chapter 5 a new way of fitness evaluation of models is introduced that exploits the data balancing methods of Chapter 2 with a modified definition of prediction accuracy for imbalanced data. Two different methods of data balancing for GP are presented. One uses the weights reflecting the relative information content of input-output data records directly for performing regression with the weighted prediction error. The second method compresses the input-output data set to a smaller subset of similar information content and

performs standard regression on a subset of data with enhanced exploration.

In Chapter 6 an alternative strategy for model evaluation in Pareto GP is introduced. It is based on the principles of goal softening and ordinal optimization, and is designed to improve the balance of exploitation and exploration in the evolutionary search.

In Chapter 7 a new method for incremental evolutions on essential data records is presented. It combines ideas of data balancing (from Chapter 2) and goal softening (from Chapter 6) into one new framework. In this framework, the data records ordered according to decreasing importance are added incrementally to the modeling system, starting from a very small subset with very large population size. The population size decreases in the course of evolution as the training subset size increases, to keep the computational budget constant per iteration.

Statistically significant performance improvements of these methods as compared with the standard Pareto GP are observed on a number of various regression case studies.

Part III of the thesis is devoted to Problem Analysis and Reduction. It assumes that the models developed in Part II are carefully scrutinized, interpreted, and validated for drawing preliminary conclusions on the difficulty of the modeling problem. Part III consists of Chapter 8 and presents some practical aspects of using symbolic regression results for estimating problem difficulty, in particular for variable selection, convergence identification, trustworthiness evaluation, and adaptive data collection.

Contents

1

Introduction into Model-Based Problem Solving

1.1 Motivation for data-driven modeling

Models lie at the core of any technology in any industry, be it finance, manufacturing, services, mining, or information technologies. The competitiveness of modern economy and the growing urge for sustainability call for a change in the industry and therewith in the underlying models. Rapid creation of new and 'better' models is a convenient aid to guarantee the long-term value, maximize efficiency, and minimize the costs.

Industrial processes are driven by fundamental models that describe (or aim to describe[1]) the true laws of nature. Comprehension and development of accurate fundamental models is one of the primary goals of scientific research. Control over these models or at least awareness on what are their driving variables is one of the primary goals of business, since it allows selecting optimal strategies for maximizing impact and profit.

Fundamental models that accurately describe complex systems and processes without severe simplifying assumptions are generally unknown. Those models that exist (and are often called fundamental models) either are of limited applicability in cases where their (fundamental) assumptions are violated, or are

[1] For non-deterministic processes, e.g. stock markets, they have to describe both laws of nature and artifacts.

of limited trustworthiness due to the lack of empirical validation. Both limitations put unavoidable expectations on the development and exploitation of empirical modeling techniques, that use observations (data) to model the processes.

Polymer manufacturing, for example, aims to describe and control kinetic and structural properties of polymers based on first principles of physical and chemical processes. This is a challenging task since polymers essentially change their behavior and even physical structure under different environmental conditions, and also because laws of mechanics do not describe all aspects of such changes. The constitutive equations describing the behavior of polymers are generally unknown and frequently involve unknown functions even when simplified kinetic models are used. The form of these fundamental models may be so complex and different from classical forms, that the relevant mathematical apparatus is not yet available to solve, or even describe them fully. In these cases, the available data can be used for empirical inference of the fundamental relationships between the process variables and for accelerating the research.

Quantitative asset management is another example of a discipline that actively exploits new empirical modeling technologies for validation and correction of existing theoretical models. Quantitative asset management aims at modeling efficient stock markets for a purpose of deriving optimal strategies for stocks and making profit either by selling over-priced stocks, or buying under-priced stocks. The traditional theoretical models of asset pricing, like the Capital Asset Pricing Model by Sharpe (1964) or arbitrage pricing theory by Ross (1976) are linear and have limitations in describing complex non-linear information-rich markets (see (Becker et al., 2006, 2007)). In this case advanced empirical modeling methods (and genetic programming in particular) are used to derive non-linear relationships between a multitude of potential factors and increase reliability of forecasting by quantitative models (see (Caplan and Becker, 2004; Neely and Weller, 2003; Neely et al., 1997)).

Another example of using (non-linear) empirical models is forecasting recessions in macroeconomics. Several studies report that recessions can be predicted through simple financial variables like interest rates, stock price indexes, monetary aggregates, see e.g. (Dueker, 1997; Estrella and Mishkin, 1996) reporting that the slope of the yield curve (the spread between the interest rates on the Treasury notes) is a good predictor of recessions of the U.S. economy over one quarter periods. These studies use so-called probit models to estimate the

probability of recession as a binary variable through a selected set of financial indicators (see also (Bernard and Gerlach, 1998) and (McPhee and Miller, 1995) for recessions in Canada). Probit models are linear in their parameters and use the assumption that each input variable plays the same role in all recessions over a selected period. Recent studies increase the accuracy of recession forecasting through the use of non-linear empirical modeling methods such as multivariate adaptive regression splines (see (Sephton, 2001)).

This thesis focuses on the development of non-linear empirical models describing the relationships in given data. We use the terms empirical and data-driven as if they were the same. Data-driven models are built using observed and captured properties of the system, exhibited under different conditions or over time, and expressed in the form of data. In this study numeric real-valued data are used for model development, and no assumptions on data distribution are made.

The task of empirical modeling, or data modeling lies in using a limited number of observations of systems variables (data) for inferring relationships among these variables.

The system under study is characterized by a number of control parameters, which are usually difficult to monitor, e.g. they can be measured only in a lab, or can be simulated only at high time- and computational costs. Empirical modeling attempts to express these critical control variables via other controllable variables that are easier to monitor, can be measured more accurately or timely, are cheaper to simulate, etc. Control variables are referred to as outputs, or responses, or response parameters, or key performance indicators, etc. Variables, or properties, that are used for expressing the response are called inputs, or input variables, design parameters, or predictors.

An example of a data matrix is given in Figure 1.1. A combination of values of all input variables and the associated values of the output variables is called a data record, data point, design point, fitness case, or scenario. Data coming from the measurements may have missing values at any place in the data matrix, or be wrong due to noise or measurement errors. The simulated data may have gaps in the response values, corresponding to the infeasible realizations of certain combinations of input variables. The task of the modeling is to detect the driving input variables that cause the change in the observed response variables, and formulate the exact relationship in a form of an **accurate** model. The quality of

Figure 1.1: Data is the input of a data-driven modeling process. The table below is a schematic view of the data matrix, where each line corresponds to a data point, also known as sample, data record, fitness case, data level.

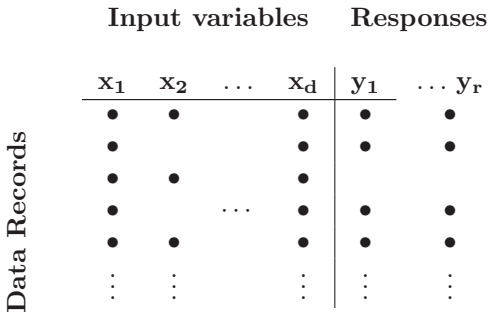
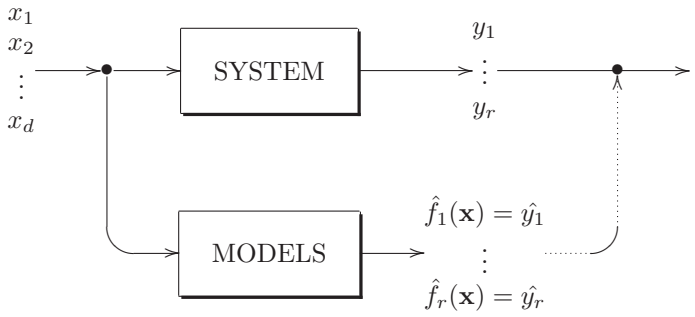


Figure 1.2: The goal and the output of a data-driven modeling process is a MODEL **or a collection of** MODELS **that approximate the behavior of the observed** SYSTEM**.** Based on the fixed values of the observed input variables, the model should generate an accurate prediction $\hat{f}$ of the output, such that it is close to the observed output y with respect to a selected fitness measure.



this model is assessed by the resemblance of the predicted output to the observed output based on a number of data points.

Figure 1.2 presents a schematic view of using modeling for prediction. SYSTEM is the system under study (it can be a real process, or a black-box simulation), $(\mathbf{x}_1, \ldots, \mathbf{x}_d)$ is a set of input variables, $(\mathbf{y}_1, \ldots, \mathbf{y}_r)$ is a set of response variables, and MODEL is a collection of modelled relationships predicting observed responses $(\mathbf{y}_1, \ldots, \mathbf{y}_r)$: $\{\hat{\mathbf{y}}_j = \hat{f}(\mathbf{x}_1, \ldots, \mathbf{x}_d),\ j = 1 : r\}$.

For many industrial applications the resulting relationships between the input

variables and the output variables of a physical system can be found implicitly, e.g. in a form of a simulation tool (see, for example, (Kleijnen, 2005; Kleijnen et al., 2005)). Such an implicit[2] relationship is called a black box. Since black-box simulations are usually developed to simulate processes of extreme complexity, they themselves are complex and computationally intensive. Computing the predicted value of the response based on the given values of inputs may take a long time and severely limits the possible number of calls to the simulation tool. Furthermore, black box models are difficult to invert. Lacking insight into the true input-output relationship makes it impossible to answer simple questions about the response, e.g. what are the ranges of input variables that cause the response to take certain values, not necessarily optimal? To gain insight into an implicit black-box model and to ease its interpretation and manipulation, we can create explicit models that mimic its behavior and relate black-box inputs and outputs. This process is called meta-modeling. In this thesis no assumptions are made about the source of data or its noise distribution. In fact, all case studies in this thesis are based on modeling given (historic) data, with data either sampled from explicit equations. simulations, or taken from measurements of real processes.

1.2 Requirements for empirical models

The requirements for empirical models in an industrial setting are defined as follows (see (Kotanchek et al., 2003; Smits and Kotanchek, 2004; Vladislavleva et al., 2008)):

1. Capability for (a) on-line **reliable prediction** of process outputs within the given range of operating conditions and *outside* this range, and (b) trustability metric for these predictions.

2. **Interpretability** and possibility to integrate information from fundamental models.

3. **Low development and maintenance cost** with little or minimal operator intervention.

[2]By 'implicit' we mean anything that differs from an equation containing the output variable and a subset of input variables, expressed analytically

4. **Robustness** with respect to the variability in process inputs.

5. **Adaptability to novelties** in data and tunability toward changes in the process.

Since both measured and simulated data are very often corrupted by noise, and in case of real measurements can be driven by a combination of both measured and unmeasured input variables, empirical models should not only accurately predict the observed response, but also have some extra generalization capabilities. The same requirement holds for models developed on simulated data. Examples of such capabilities are insensitivity to a certain amount of noise in the inputs or a capability to extrapolate the response outside the observed input region (in fact, in a multi-dimensional data space, any deviation from the training points has to be performed with caution and will often be referred to as extrapolation[3]). We also would like models to exhibit not only required properties, but also additional convenient properties like compactness, small number of constants, etc. It is important, that generated models are interpretable and transparent, in order to provide additional understanding of the underlying system or process.

1.3 Data-driven modeling methods

There is a multitude of empirical modeling techniques that are used for constructing input-output regression models, but we wish there were one technique, producing models that fulfill all the requirements listed above.

[3]In general, interpolation is defined as prediction of an unknown function *within* the input range of a finite set of samples, and extrapolation is defined as prediction *outside* the input range of data samples *or* with higher uncertainty. In this thesis we consider modeling high-dimensional undesigned data, which may be imbalanced, i.e. may not have a uniform density in the input space. For imbalanced, high-dimensional data sets, however, the standard definitions of interpolation and extrapolation are not always appropriate. If the input points of a one-dimensional modeling problem are located on the intervals $[0, 1]$ and $[10, 11]$, we consider prediction at point 5 to be an extrapolation, despite the fact that 5 belongs to the convex hull of the input samples. In general, if the data is not uniform in the input space, it may consist of several connected components (clusters), which number depends on the choice of the user. The decision on whether we extrapolate or interpolate will depend on the location of the prediction point relative to the clusters, while the uncertainty of prediction will depend on the data density in the neighborhood of the prediction point, and on how well the point is surrounded by its neighbors in the input space. Since the data density decreases when the number of dimensions increases at the fixed number of data samples, the uncertainty of prediction grows in higher dimensions, which substantiates the usage of the term extrapolation.

We mention several most common methods to produce empirical models: linear regression (Box and Draper, 1986), nonlinear regression (Johnson and Wichern, 1988) including multi-variate rational interpolations on points (Cuyt and Verdonk, 1985) and on vertical segments (Salazar Celis et al., 2007) and multi-point approximations of response surfaces for optimization (Toropov, 1989), moving least-squares (Choi et al., 2001; Levin, 1998), kriging (Kleijnen, 2008a; Sacks et al., 1989), multi-variate adaptive regression splines (Friedman, 1991), radial-basis functions (Powell, 1987), neural networks (Haykin, 1994), support vector regression (Cherkassky and Mulier, 1998; Vapnik, 1997), genetic programming (Koza, 1992), and particularly Pareto Genetic Programming (Smits and Kotanchek, 2004).

Modeling practitioners widely accept that there is no universal approach to empirical modeling, and that the future lies in combining working principles from various approaches and blending them into hybrid methodologies. This section makes a modest attempt to discuss the main differences between some of the mentioned methods, to look for complementary principles that should be used together in modeling complex industrial processes and to locate the capabilities of the Pareto GP method (which is the main focus of this thesis) on a general map of possibilities of regression techniques.

Several studies (Jin et al., 2003, 2000; Shyy et al., 1999; Simpson, 1998) attempting to compare the different approaches to data-driven modeling agree that no definite conclusions can be drawn on the general advantage of any particular technique over others. The difficulty is explained by multiple competing criteria for comparison, which tend to vary wildly over different techniques.

We categorize the approaches into parametric and non-parametric in the following way[4]. A parametric approach to regression consists of fixing a model structure *a priori* and finding optimal values for the coefficients (parameters, or weights), that minimize a certain criterion, usually the sum of squared errors. Polynomial regression is a classical parametric modeling method. E.g., for a second-order model on d input variables, the approximating function is selected to be the following:

[4]This division is different from the one used in statistics, where parametric methods assume certain distribution of samples

$$\hat{f}(\mathbf{x}_1, \ldots, \mathbf{x}_d) = \beta_0 + \sum_{1 \le i \le d} \beta_i x_i + \sum_{1 \le i \le j \le d} \beta_{i,j} x_i x_j, \tag{1.1}$$

where β_0,β_i, and $\beta_{i,j}$ are the unknown coefficients.

Polynomial regression is a linear parametric method, since it involves solution of a linear system of "normal" equations in the unknown parameters.

Rational interpolation is a non-linear parametric method, since the model structure is fixed to represent a ratio of two polynomials, and quadratic programming is used to estimate optimal values for unknown coefficients. We address two kinds of rational interpolation - the standard multivariate rational point interpolation, and rational interpolation on intervals (Salazar Celis et al., 2007). The latter is a new method to construct rational interpolations through intervals of a certain length associated with each data point. The length of the interval determines the maximal tolerable interpolation error (insensitivity zone). The technique seems to be an elegant extension of rational point interpolations, which removes the dependence of model complexity (measured in the number of terms) from the data size, and allows a more flexible modeling of data sets with a large number of records.

Feed-forward neural networks (NNs) presented as multi-layered perceptrons build the multi-variate prediction model as a superposition of weighted combinations of unary basic functions (called inner nodes) and input variables. Neural networks defined as (ordinary) radial basis function networks (RBFNs) build prediction models as a weighted combination of radial basis functions with centers in the input points. The generalized RBF networks use a much smaller number of basis functions (inner nodes) at the expense of having to estimate the optimal location of their centers in addition to unknown weights. We categorize neural networks as a non-linear parametric method.

Support vector machines (SVMs) are non-linear parametric models originating from statistical learning theory (Cherkassky and Mulier, 1998; Vapnik, 1982). A support vector machine is mapping the data space into a multi-dimensional feature space, where the prediction function $\hat{f}_{Feat}$ is *linear* in the unknown parameters $\mathbf{w}$ and b: $\hat{f}_{Feat}(\mathbf{x}, \mathbf{w}, b) = \mathbf{w}\mathbf{x} + b$. The type of transformation $\phi : \mathbf{x} \mapsto \phi(\mathbf{x})$ into the feature space is chosen *a priori* and is usually polynomial, RBF, or a combination of these two (Jordaan, 2002).

SVMs are said to be characterized by their kernel function, denoted as

$K(\mathbf{x}, \mathbf{x_0}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{x}_0)$. The optimal weights for the prediction function are computed through minimization of an ε-insensitive error function:

$$\|\mathbf{w}\| + C \sum_{i \leq N} max\{0, |y^i - \hat{f}(\mathbf{x}^i)| - \varepsilon\}, \tag{1.2}$$

where ε is a tolerable error, C is a regularisation coefficient, and $\mathbf{w}$ are the weights of the $\hat{f}$. Since weights are computed for the feature space, all $\mathbf{x}$ are transformed into the $\phi(\mathbf{x})$ during the optimization of $\mathbf{w}$ and b.

The difference between SVMs and NNs is that a NN method defines the model structure first and evaluates optimal coefficients for it such that the prediction error is minimized. A SVM method first determines the tolerable generalization error (and also a type of kernel and the regularization term), and then searches for the optimal structure for the model, i.e. weights for 'optimal' data records, called support vectors. The theoretical advantage of the SVM method over NNs is that the error function has no local optima, and the generalization error does not depend on the dimensionality of the data space by design (Jin et al., 2003). The main difference between support vector regression and other approximation techniques is the fact that the SVM model is a not a direct function of the input variables, but a function of data points (support vectors), and has the following form: $\hat{f}(\mathbf{x}) = \sum_{k \in SV} \omega K(\mathbf{x}^k, \mathbf{x}) + b$, where SV is a smallest subset of points with positive optimal weights called support vectors, and ω and b are the unique estimated parameters. This is the main hurdle for the interpretation of SVM models, since such notions like variable interactions are absent by design. A strong point of the SVM method besides a strong theoretical foundation, is its ability to detect outliers in the data; see (Jordaan, 2002).

All methods discussed in this section are global, in the sense that they produce a prediction of the observed response globally on the entire training range. We will, however, divide parametric methods into global and local ones, depending on the global or local nature of their parameters. For example, polynomial regression is a global parametric method whose coefficients do not depend on the point at which the predicted response is evaluated.

Moving least-squares (MLS), is a local parametric method, since it requires re-calculation of coefficients when the evaluation point changes. The difference between the polynomial moving least-squares model and a standard polynomial regression model obtained via least-squares is that the coefficients of the MLS

model are computed *locally* for a point $\mathbf{x}$ through the least-square minimization of the *weighted* error function:

$$\sum_{i=1}^{N}(\hat{f}(x^i) - y^i)^2\theta(\|\mathbf{x} - \mathbf{x}_i\|), \tag{1.3}$$

where θ is a non-negative weight function, $\|\cdot\|$ is a selected norm in the input space $\mathbb{R}^d$, usually Euclidean, and $(\mathbf{x}^i, y^i)$ is the i-th record of the data set. Weight θ is analogous to both the neighborhood size and insensitivity zone - the MLS approximation can be made local, if $\theta(p)$ quickly converges to 0, when $p \to \infty$, and interpolation with MLS can be achieved when $\theta(0) = \infty$.

Kriging is another local parametric method for building global multi-variate approximations, in fact, interpolations. A kriging model is determined as a sum of a global predictor, often called a signal function $S(\mathbf{x})$, and a localized 'deviation' $\zeta(\mathbf{x})$:

$$\hat{f}(\mathbf{x}) = S(\mathbf{x}) + \zeta(\mathbf{x}), \tag{1.4}$$

where $\zeta(\mathbf{x})$ is normal random function with zero mean and non-zero covariance: $Cov[\mathbf{x}^i, \mathbf{x}^j] = \mathbf{R}(\mathbf{x}^i, \mathbf{x}^j)$, where $\mathbf{R}$ is a correlation function selected from a family of parametric correlation functions, often as $\mathbf{R}_{\mathbf{q},\theta}(\mathbf{x}^i, \mathbf{x}^j) = exp[-\sum_{k=1}^{d}\theta_k|\mathbf{x}_k^i - \mathbf{x}_k^j|^{q_k}]$ ($\mathbf{x}_k^j$ if the k-th component of a sample point $\mathbf{x}^j$). Prediction of the final kriging model in point $\hat{\mathbf{x}}$ is determined by a weighted sum of observed responses $\hat{f}(\hat{\mathbf{x}}) = \sum_{i=1}^{N} c(\hat{x})y^i$, where the optimal values minimize the mean squared prediction error. Coefficients θ_k and q_k are usually obtained via the maximum likelihood estimation method. Computation of coefficients involves inversion of the matrix of size $d \times d$ (d is the dimensionality), and hence, puts limitations on using kriging for very large data sets.

Multivariate Adaptive Regression Splines (MARS) is a flexible regression technique with a self-explanatory name: the final model is obtained as a weighted linear combination of separate splines (or basis functions) fitted to distinct intervals of the input variables. The splines are obtained using a non-parametric method, but since they are combined in an optimally weighted sum, it is fair to call this method semi-parametric. The MARS algorithm exhaustively searches over all possible locations of the endpoints of spline interpolation, as well as all possible variables and variable interactions. The knots, variables and interactions

are optimized simultaneously to minimize the error function, which changes adaptively to improve the model at each step of adding a new spline. Once MARS determines the optimal number of basis functions (splines) and knot locations, a final least-squares regression provides optimal coefficients for their weighted linear combination.

Genetic programming (GP) is a stochastic iterative search technique, that navigates in the space of all possible symbolic models defined as valid mathematical expressions on the given set of input variables, basic functions, and constants, and searches for a set of models optimizing a fitness objective or a trade-off of competing objectives (such as prediction accuracy of the training set and model expressional complexity in Pareto GP); for genetic programming of which a number of variants exist see the books by Banzhaf et al. (1998); Koza (1992, 1994); Langdon and Poli (2002); O'Neill and Ryan (2003); Poli et al. (2008), for a multi-objective variant studied in this thesis see (Smits and Kotanchek, 2004), the next section, and Chapter 3 for more details. Unlike the MARS method, which *exhaustively* searches for appropriate knots, variables, and interactions using the data matrix, genetic programming exploits *evolutionary search* to find appropriate model structures in a huge (in general infinite) space of possibilities. Despite the vastness of the search space, GP has been shown to discover regression models that are both compact and sufficiently accurate in a number of industrial applications for problems with a large number of nuisance variables, noise, and correlated inputs. One of the best capabilities of GP is automatic identification of significant inputs, which implies automated pruning of nuisance inputs and increased applicability of final models built upon significant variables only. More benefits and potential pitfalls of GP (with a focus on Pareto GP) are given in the next section.

The majority of these techniques for empirical modeling have evolved into separate disciplines, with solid bibliographies, software packages, and extensive research *within* each discipline. Table 1.1 provides a short and simplistic comparison of the capabilities of the methods with respect to the set of (self-explaining) questions of interest. The main purpose of this table is to obtain a snapshot of differences among the methods and to identify the points of potential synergetic co-application.

The question on scalability of a method is not raised in Table 1.1. The problem is that the notions of scalability and data quality are used very loosely among

Table 1.1: Part I. A draft of a high-level comparison of common empirical modeling methods.

Feature	Linear Regression	Rational Interval Interpolation	NNs	SVMs	Kriging	MLS	MARS	GP & Pareto GP
Knowledge about explicit model structure required?	Yes	No	No	No	No	Yes	No	No
Parametric or Non-parametric	Param.	Param.	Param.	Param.	Local Param.	Local Param.	Semi-param.	Non-param.
Possibility for Local Adaptation	No	No	No (Yes for RBF Ns)	No	No	No (Yes for some θ)	Yes	No
Model complexity depends on the # of data samples	No	No (Yes for PI)	No (Yes for RBFNs)	No	Yes	No	No	No
Potential to create compact models irrespectively of data size and structure	High, if structure is right	Moderate, if interval is large (No, for PI)	Limited	Limited	Limited	High, if structure is right	Moderate	High
Can interpolation be guaranteed if needed?	No	Yes	No	Almost, if $\varepsilon = 0$ & $C = \infty$	Yes	Almost, if $\theta(0) = \infty$	No	No
Danger to over-fit the data without explicit complexity control?	No	Reduced, if interval is large	High	High	High	Reduced	Reduced by design	High
Can final models provide insight into the problem?	Yes	Hardly	Hardly	Hardly	Hardly	Hardly	Hardly	Yes

Table 1.1: Part II. A high-level comparison of common empirical modeling methods.

Feature	Linear Regression	Rational Interval Interpolation	NNs	SVMs	Kriging	MLS	MARS	Pareto GP
Complexity control possible?	Yes, by fixing a model structure	Yes, by increasing the interp. interval	Yes, by limiting # of hidden nodes	Yes, by fixing the penalty term, increas. ε-zone or changing K	Yes, by using a subset of data	Yes, by fixing a model structure	Yes, by using a subset of data, partly present by design	Yes, by explicit control of model properties
Implicit assumption that input variables are independent?	Yes	Some, otherwise numerical problems possible	No	No	Some, otherwise correlation should be redefined	Yes	No	No
Implicit assumption that all input variables are significant?	No	No	No	Yes	No	No	No	No
Danger of having insignificant variables in final models	Present	Present	Present	Present	Present	Present	Present	Not Present, or Heavily Reduced
Danger of NOT having significant variables in final models	Not Present	Not Present	Present	Not Present	Not Present	Present	Present	Present
Major practical challenge	Choosing right model structure	High dimensionality of data	Over-fitting	Insignif. inputs	Large # Records in data	Choosing right model structure	Large data matrix	Huge search space & stochasticity

disciplines and depend on the application area. For example, in some industrial modeling projects carried out during the PhD research, the data with 20 input variables, and 25 measurements per variable (in modeling for new products), or with 10 input variables and 30 measurements per variable (in developing new measurement techniques) were given as very rich data sets.

The evolutionary search was developed for handling tremendously difficult problems, which are intractable for classical approximation techniques. When we talk about scalability of Pareto GP or other evolutionary techniques to large data sets, we talk about handling hundreds to thousands of variables, and hundreds to millions of records.

For rational interpolation, for example, data sets with dimensionality 8-10 are already high-dimensional. It does not make the method any weaker, it only suggests that one should probably not use rational interpolation if the training data contains 50 variables and 8000 records. It is possible, however, (and is often the case for real-life problems) that only a handful of those 50 variables or low-order transformations on some variable combinations are significant, but we do not know which, and the correlation analysis or the principal component analysis do not provide such information. In this case it is beneficial to perform several GP runs on the training data to identify driving variables, or combinations, extract them, and make the problem tractable for rational interpolation, since the latter can successfully impose some *a priori* constraints on the behaviour of the final model.

In general, it will be interesting to express the scalability of the methods as a function of the number of available data records N, and as a function of the number of input variables d[5].

A discussion on Table 1.1 with a focus on Pareto GP continues in the next section.

[5]This and many other valuable suggestions on improvement of Table 1.1 were made by Trent McConaghy. Trent also suggested to add Random Forests and Boosting Regression Trees methods to the Table, as well as distinguish between piece-wise linear and piece-wise non-linear methods instead of using one MARS column, and separately consider a good second-order polynomial models and high-order polynomial models within linear regression. These adjustments will appear in the on-line version of the thesis.

1.4 Symbolic regression via GP: benefits and challenges

1.4.1 The task of symbolic regression

One of the best descriptions of symbolic regression via genetic programming (SR via GP) is given by Schmidt and Lipson (2006), who wrote that " unlike polynomial regression or machine learning methods which minimize the error, SR is a system identification method." The challenging task of symbolic regression is to identify a *convincing* model $\hat{f}$ (or a set of models), that gives a prediction for the response variable and is expressed analytically using the minimal set of input variables and the given set of basic operations and constants.

Industrial challenges we are attempting to solve have two main characteristics: (1) No or little *a priori* information is known about the underlying system, and, therefore, no assumptions about models can be made; (2) the available data is high-dimensional with either an abundant or an insufficient number of samples.

No information about model structure, unknown distribution of data samples, and 'bad' data sets dramatically reduce the number of alternative techniques to start empirical modeling. If the task is structure identification, then parametric methods should be excluded from the number of options to perform this identification. Considering the alternatives presented in Table 1.1, this leaves multivariate adaptive regression splines and GP as the only non- and semi-parametric methods. If we want models that produce reliable predictions outside the training region (see requirement (1) of Section 1.2, we certainly want extrapolation to be influenced by the behaviour of the entire (true) response surface, and not only by the neighboring splines in case of MARS. The need to produce a reliable extrapolation of the response, predicted by an identified non-parametric model, suggests to put aside MARS and settle on genetic programming. We are going to do this for the rest of the thesis, and focus on Pareto GP, a genetic programming with archiving and multi-objective fitness function. An overview of other variants of GP can be found in a recent monograph by Poli et al. (2008).

1.4.2 Challenges in view of requirements to models

Genetic programming, and evolutionary search in general, is a very flexible methodology, allowing vigorous application and co-application of all kinds of strategies and meta-strategies, as long as a cycle between model generation & variation, model evaluation, and model selection is maintained. Having settled for a particular method for constructing input-output regression models and presuming that it is very flexible, we will come back to the requirements to good models formulated in Section 1.2 and create a list of necessary features that our GP-based system should possess in order to produce models satisfying those requirements.

(1) *Capability for on-line reliable prediction of process outputs within the given range of operating conditions and outside this range:* The need for reliable predictions *within* the training range imposes the use of prediction error as one of the optimization criteria for model development. The need for reliable predictions *outside* the training range, i.e. in the unobserved areas of the data space imposes the need to produce models with smooth response surfaces and minimized potential for pathological behaviour under extrapolation.

(2) *Interpretability and possibility to integrate information from fundamental models:* Interpretability is a tricky notion. It is subjective, and can be said to solely depend on the creativity of the observer. However, in reality, it is very easy to say whether the model is interpretable or not. It is at least always easy to tell whether the model is uninterpretable. By a potential for interpretability or for insight I will mean the potential to connect observations of a model (not only model's prediction error) to hypotheses about the underlying problem. For example, if model $\hat{f}(x_1, x_2, x_3, x_4, x_5) = \left(x_2 + \frac{x_3}{x_4}\right) e^{1-\left(\frac{x_3}{x_4}\right)^2}$ has a sufficiently high prediction accuracy, it suggests that variable x_5 is insignificant, and transformation $\frac{x_3}{x_4}$ can be an important predictor of the response.

The possibility to integrate information from fundamental models is an open area of research. Pareto GP and GP in general are open for inclusion of various meta-variables or important basic operators (e.g. taken from fundamental models) into the search process, but property preservation,

like e.g. creation of monotone models, or only square-integrable functions, is non-trivial and has to involve either a modification of the fitness function, or introduction of new property-preserving modification operators. This thesis does not contain a study on imposing *a priori* constraints to GP solutions during the modeling process. It is however worth mentioning that Pareto GP is designed to generate rich sets of diverse final solutions, which gives the user the opportunity to explore these sets of solutions *a posteriori* in order to select models with desired properties, or the ones that agree with intuition.

(3) *Low development and maintenance cost with minimal operator intervention:* Low development cost implies that models should be developed with minimal time and computational effort. The ultimate goal is a one-hit-of-a-button software integrating data preprocessing, analysis and adaptation, model development and problem analysis phases. Low maintenance cost refers to deployment costs associated with implementation and maintenance of the model or a model ensemble in a real system, e.g. for predicting a hard-to-measure property of a production process. Low maintenance implies that model prediction will not require monitoring, and has a capability for self-assessment expressed in a trust metric. This is useful when, for example, processing conditions change unexpectedly. In general, for multi-variate measurements it is hard to say when a point is appropriate or is an outlier. In case of changed processing conditions, a trustworthy model or model ensemble will alarm about entering an unobserved zone, and allow focused process control.

(4) *Robustness with respect to the variability in process inputs.* The requirement for robustness is inspired by modeling data coming from real measurements, where not only the response, but also the inputs can be corrupted by considerable noise. The need for robustness under small variation of the inputs imposes a constraint on the non-linearity of the response surfaces associated with models. The smoother, the more robust (most of the time).

(5) *Adaptability to novelties in data and tunability toward changes in the process:* The need for adaptability requires the system to be capable of identifying the novelties in data, assessing whether they correspond to regime changes or to measurement outliers, or require re-modeling. As mentioned in Table

1.1, GP does not possess a capability for local adaptation, i.e. if a new sample point is added to the data set, in principle, a total re-modeling of the data is required. It is, however, possible to supply an initial set of alternative solutions with results from the previous efforts and (possibly) make the revised model discovery more efficient.

Capabilities mentioned above can be considered as generic research goals for the development of a practical modelling tool. Before we zoom in on the research goals of this thesis, we provide a short description of the Pareto genetic programming method, and evaluate its benefits and challenges with respect to the above-mentioned research goals.

1.4.3 Genetic programming versus Pareto GP

Pareto genetic programming is a branch of the genetic programming methodology (GP). GP for symbolic regression is an iterative search technique that looks for appropriate expressions of the response variable in a space of all valid formulae containing some of the given input variables and some predefined functional operators, like summation, multiplication, division, exponentiation, inversion, sine, etc.

At each iteration step the genetic programming system considers a sufficiently large quantity of various formulae, selects a subset of the 'best' formulae according to certain criteria of goodness, and (re)combines these best formulae to create a rich set of new formulae for the next step. This approach is inspired by principles of natural selection, where offspring should inherit good features from both parents or undergo a beneficial mutation to be successful in survival, adaptation and further propagation. For more general information about genetic programming we refer to the monographs by Banzhaf et al. (1998); Koza (1992, 1994); Langdon and Poli (2002).

The rationale of doing the 'evolutionary' search in a huge space of alternatives is to balance the exploitation of the good solutions that are found so far, with exploration of the new areas of the search space, where even better solutions may be hiding.

Exploitation usually happens by recombining the features of models that are relatively good, e.g. they both predict the observed response with a reasonably low errors. The fundamental expectation here is that the recombination of good

features encourages the overall improvement of newly created models, despite the fact that such improvement is not at all guaranteed.

Exploration of new areas of the solution space happens by introducing changes in the individual models. Such changes can be minor, e.g. if we decide to substitute the multiplication operator by a division in the formula expression, or replace an input variable by another variable; or major, e.g. when we decide to randomly re-initialize the model. Part II of this thesis describes the processes of model generation, selection and evaluation for symbolic regression via Pareto GP in greater detail. It is important to note, that recombination of sub-parts of two individuals can often be an explorative operator when, e.g., used to cross-over high-quality individuals with randomly generated individuals. In such cases recombination is acting as a macro-mutation, aimed at speeding-up the pace of exploration.

There is a small and yet crucial difference between Pareto GP and a classical GP. Pareto GP is an archive-based search method, that actively exploits and maintains the new entity during the evolution - the archive of the 'best' discovered so far individuals. The second difference is that the concept of what is 'best' is multi-objective in Pareto GP. This means that there are at least two or possibly more criteria used for selecting 'good' individuals for further propagation. Often these criteria are prediction error and model expressional complexity. Since these optimization objectives are competing (there are always ways to achieve the same prediction error at the expense of greater structural complexity), the performance of individuals is compared with respect to the Pareto-dominance relation in the objective space of model complexity and model error (see Chapter 2). In Pareto GP, model development happens in parallel with automatic identification and exploitation of driving inputs that influence the observed output (Smits et al., 2005).

Pareto GP with a presence of the archive and the two-objective optimization of prediction error and model complexity has been shown to significantly outperform the classical GP approach, which uses a single objective minimization of prediction error for model selection on a variety of test problems see (Kotanchek et al., 2006; Smits and Kotanchek, 2004). The major impact of Pareto GP is in the fact that (1) it allows interpretability of it solutions (since the drive for minimization of structural complexity of models is explicitly introduced into the search), and (2) it produces diverse solutions by design (because the

Pareto front models, consisting of optimal trade-offs in complexity versus error space are always present in the archive of solutions). Explicit complexity control also facilitates generation of simplified expressions, and hence contributes to improved generalization capabilities of solutions (for the same accuracy, models with minimal complexity are preferred, and the other way around). In Chapter 3 we talk more about the fact that simplicity does not always imply generality, however, it is important to note, that expressional complexity control implicitly mitigates the risk of numerical inaccuracies, since e.g. expression $x_1 + x_2$ will be selected over $x_1 + Exp[Ln[Exp[Ln[x_2]]]]$.

The fact that symbolic regression via GP does not impose any assumptions on the structure of the input-output models means that the model structure is to a large extent determined by data[6].

On one hand, this is an advantage and a unique capability of symbolic regression via GP over the other global approximation techniques (see, e.g., Table 1.1 for parametric and non-parametric methods). Since the search space is not limited to expressions of a certain structure, there is a potential to develop inherently simpler models, than, for example by linear regression on polynomials or spatial correlation analysis, where the number of terms in the model depends on the size of the data set. It is for instance very handy to have a sine function among the basic operators to model the response generated by, e.g., a sinc function, compared with a challenge of figuring out the structure and the coefficients for its series expansion, which approximates $sinc(x)$ accurately enough[7].

On the other hand, the absence of constraints on model structure is the greatest challenge for symbolic regression, since it vastly increases the search space of possibilities, which is already inherently large. Obviously, larger data sets with more input variables and more records, make symbolic regression even harder, since a larger space of possible models has to be explored, and larger computational effort has to be applied to evaluate predictions of potential solutions on more records.

[6] A note for GP researchers: Recent studies by Keijzer and Foster (2007); Poli et al. (2007) show that the standard random uniform crossover can and does impose a bias on the structure of tree-based models, which means that other forms of crossover should be used for recombination to justify the statement above.

[7] There is a general caution, however, for using trigonometric functions among the basic operators- they should be used only if their presence in final models is plausible from the point of view of *a priori* knowledge.

1.4.4 Research questions of this Thesis

As stated in Table 1.1, time-to-solution, or, more accurately, time-to-convincing-solutions is a major practical challenge of Pareto GP (and GP in general). And yet, an amazing observation that have been being made continuously over this PhD project, is that despite the horrendous complexity of the search space, the simple evolutionary principles do bring ParetoGP to solutions (for *all* practical modeling problems that the author happened to be involved into). These are inspiring news — if we give the system enough time and computational budget - the system will generate the solutions. What we want is to modify the system in such a way that the computational time and effort are minimized and still lead to solutions of sufficient quality.

Besides the time-to-solution, which not only refers to the efficiency of the method, but also to its effectiveness (looking for a shorter path to solutions in the search space is not the same as optimizing efficiency), there are several additional challenges in Pareto GP. Below we formulate all challenges as research questions:

1. How to minimize the computational effort needed to discover regression models of sufficient quality?

2. What are the alternatives for a more intelligent navigation through a vast search space of potential solutions, which would combine abundant exploration of new alternative solutions with sufficient exploitation of already discovered, potentially beneficial, sub-structures?

3. The search process is stochastic and requires multiple replications for confident interpretation of the obtained solutions. How to enhance model selection strategies in such a way, that it leads to robust reproducibility of results?

4. Despite explicit complexity control, solutions of the Pareto GP process obtained through simultaneous minimization of model complexity and prediction accuracy may still exhibit pathological behavior even in minor extrapolations. If structural simplicity does not always imply generality, how to control over-fitting by additional measures or strategies?

5. When modeling large data sets (with thousands of records), the Pareto GP system (as well as other GP systems) spend the majority of the computational effort on error evaluations of predictions of alternative solutions against the training set. How to enhance the model evaluation strategy in such a way that the required number of fitness evaluations is reduced to the bare minimum, while not deteriorating evolutionary progress?

6. Data sets coming from measurements can be very imbalanced. The quality of given data is intuitively deeply interconnected with the potential to develop convincing regression models, describing these data. What are the possibilities of assessing the level of imbalancedness of these data, and the ways to balance these data for the purpose of improving the results of Pareto GP?

There is one more challenge related to inclusion of domain knowledge into the modeling process. Inclusion of *a priori* information on desired model properties (like certain asymptotic behaviour, monotonicity intervals, constraints of differentials, etc.) is not yet handled in Pareto GP, but is not studied in this thesis.

Another feature of GP and Pareto GP that is important to understand is that unlike rational interpolation, kriging, or SVMs that guarantee a solution with a predefined error on the training set, the error of final solutions of Pareto GP cannot be estimated upfront. Since nothing is known about the problem and about the features of final solutions before the modeling, it is in principle unclear when to stop the modeling process. Definition of the stopping criteria and prevention of evolutionary stagnation are other open research questions. We touch these questions for Pareto GP in Chapter 8, but already indicate here that the presence of the archive in Pareto GP has strong advantages in postponing evolutionary stagnation and progress monitoring when compared with standard GP.

1.4.5 Benefits of symbolic regression via Pareto GP

Below we summarize the benefits of symbolic regression via Pareto GP over other global parametric non-linear approximation techniques and over symbolic regression via standard GP for solving hard and messy data-driven problems:

1. No prior assumptions are imposed on model structure, which gives a potential for unpredictable and counter-intuitive expressions and variable combinations to be discovered.

2. The final predicting model or model ensemble are chosen from a rich set of global non-linear empirical models that are generated automatically.

3. Sensitivity analysis of the inputs and variable selection is implicitly performed with no extra cost, so the dimensionality of the problem can be reduced.

4. No assumptions are made on the independence or significance of input variables.

5. Produced models are intrinsically more trustworthy than solutions of standard GP due to explicit complexity and non-linearity minimization (studied in this thesis) and automatic variable selection.

6. Symbolic representation of models may provide additional insight into the problem, e.g. in a form of important variable transformations.

1.5 Guide to the Thesis

1.5.1 Structure of the Thesis

During this PhD project the author spent at least half of her research time at the R&D department of a large manufacturing company. Exposure to real problems and challenges focused the research. As can be seen from the previous section, the emphasis of this thesis is put on research topics driven by the requirements to good empirical models, on those goals that potentially lead to a rapid creation of 'sellable' input-output models and to a quick understanding of the problem difficulty.

Findings of this study are presented in a practical framework of iterative model-based problem solving. We view this generic framework as an iterative feed-back loop between three stages of problem solving (just as it usually happens in real-life applications):

1. **Data Generation, Analysis and Adaptation**,

2. **Model Development**, and

3. **Problem Analysis and Reduction**.

An interesting observation is made in the "Towards 2020 Science report" edited by Emmott et al. (2006):

> "What is surprising is that science largely looks at data and models separately, and as a result, we miss the principal challenge - the articulation of modelling and experimentation. Put simply, models both consume experimental data, in the form of the context or parameters with which they are supplied, and yield data in the form of the interpretations that are the product of analysis or execution. Models themselves embed assumptions about phenomena that are subject of experimentation. The effectiveness of modeling as a future scientific tool and the value of data as a scientific resource are tied into precisely how modelling and experimentation will be brought together."

This is exactly the challenge of model-based problem solving, and researchers pursuing empirical modeling have always been aware of it[8].

Also in this thesis we attempt to bring together data, models, and problem analysis into one generic framework. Ultimately, we want to automate this iterative feed-back loop over data analysis and generation, model development, and problem reduction as much as possible, not in order to eliminate the expert, but in order to free as much 'thinking time' for the expert as possible. For successful model-based problem solving we emphasize the critical need for an expert, who will test the theory, facts (data) and their interpretations (models) against each other to iteratively develop a convincing story where *all* elements fit and agree.

The outline of the proposed framework, which also forms the structure of the thesis is presented in Figure 1.3–1.6 with the main phases explained. The rest of this subsection provides a brief description of the thesis structure, and is followed by the main contributions presented in the next subsection.

[8]at the very least it concerns the methods described in Section 1.3 of this Chapter, irrespectively of the fact that some of the methods are abused in applications they were not designed for

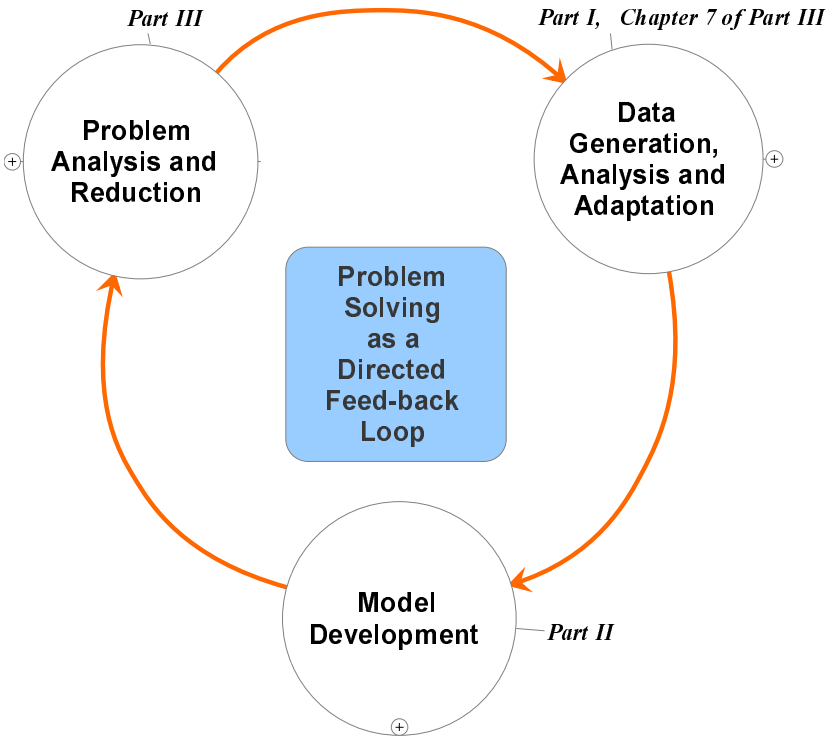


Figure 1.3: Findings of the Thesis are structured according to the generic scheme of model-based problem solving.

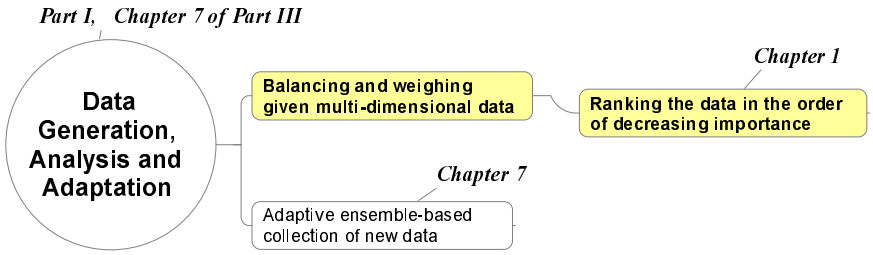


Figure 1.4: Layout of the phase of Data Generation, Analysis, and Adaptation presented in the thesis. Blocks corresponding to novel contributions are emphasized in bold.

Data Generation, Analysis, and Adaptation

Very often, especially in big companies, modelers do not have access to data creation and experiment planning. This gap is an example of a situation, where multi-variate data is given and there is no possibility to gather more or better sampled data. Modeling given data (also known as historical or legacy data) is the most common application of symbolic regression, and is related to observations of the real system.

A second situation can be distinguished, when there is a possibility to plan experiments, and to gather new observations of the response for desired combinations of input variables, but the assumption is that these experiments are very expensive, i.e. require long computation, simulation, or experimentation time. Such a situation is most common in meta-modeling for design and analysis of simulation experiments.

As Figure 1.4 suggests both situations are considered in our generic framework, however the main emphasis of the study is put on the analysis and adaptation of given data (see Chapter 2). The most recent study on Pareto GP considers exploitation of model ensembles for adaptive data collection and trustworthy symbolic regression (Kotanchek et al., 2008). The potential impact of Pareto GP on meta-modeling and adaptive design of experiments is intriguing. We touch on it in Chapter 8 and speculate on its application domain in Conclusions.

Model Development

In model development we focus on automatic creation of collections of diverse data-driven models that infer hidden dependencies on given data and provide insight into the problem, process, or system in question. Since we consider symbolic regression via genetic programming as an induction engine for input-output regression models, we present the stage of Model Development as an iterative loop over **Model Generation**, **Model Evaluation**, and **Model Selection**. The sub-components of these three stages become specific to genetic programming. However, many principles presented there are generic and can be used within other techniques for data-driven modeling, especially for the phases of **Model Evaluation** and **Model Selection**.

In Chapter 3 we summarize the basics of evolutionary search performed via genetic programming and Pareto genetic programming, and explain the stages of model generation, evaluation and selection for Pareto GP.

For effective model selection we enhance Pareto GP with an additional criterion of model simplicity and present the new strategy for model selection in Chapter 4. Also in Chapter 4 we suggest a heuristic that pursues a soft selection of models that are not only accurate, but are expressed by compact equations with smooth response surfaces.

For a more effective model evaluation on imbalanced data we study the possibilities of introducing weights to the fitness function. Variations of such 'balanced' regression are presented in Chapter 5.

For efficient model evaluation and faster learning we suggest to perform symbolic regression via Pareto GP under principles of goal softening and present two strategies to do this in Chapters 6 and 7.

Problem Analysis and Reduction

The stage of Problem Analysis and Reduction supposes that developed models are carefully scrutinized and validated, and preliminary conclusions of problem difficulty are inferred from the interpretations of the developed models. In Chapter 8, we summarize the unique advantages of using symbolic regression via genetic programming as a modeling tool for problem analysis and reduction. Many of them, such as variable selection or meta-variable detection and convergence identification, can in fact be performed on the fly during the model

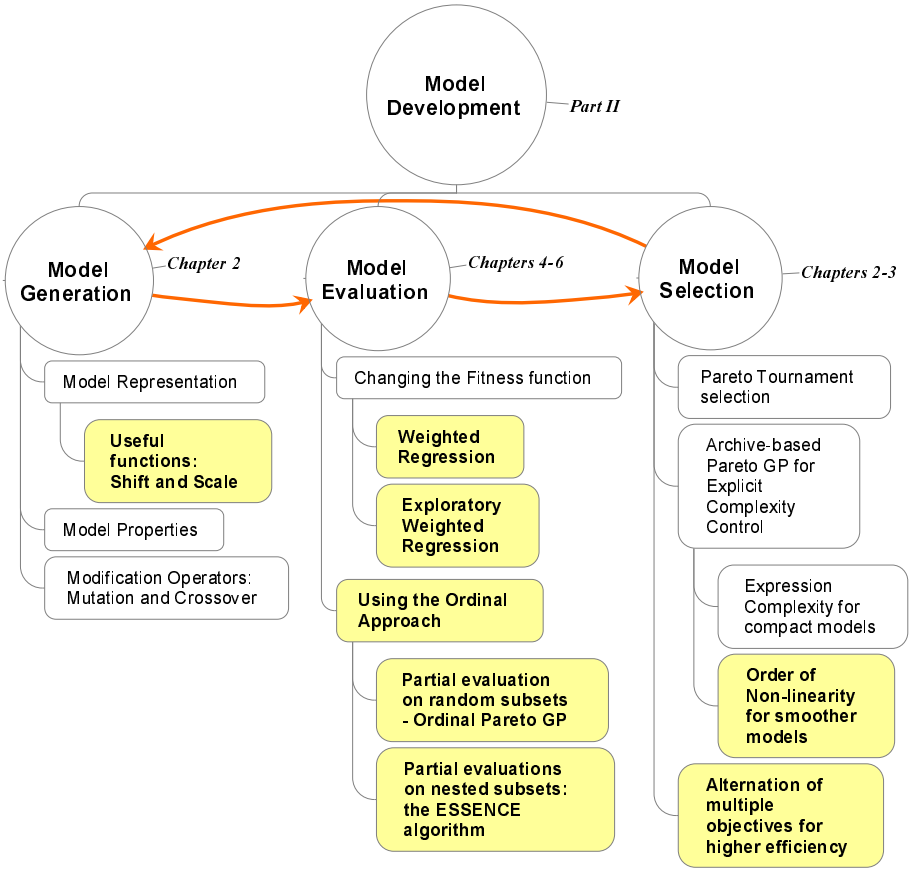


Figure 1.5: Layout of the phase of Model Development presented in the thesis. Blocks corresponding to novel contributions are emphasized in bold.

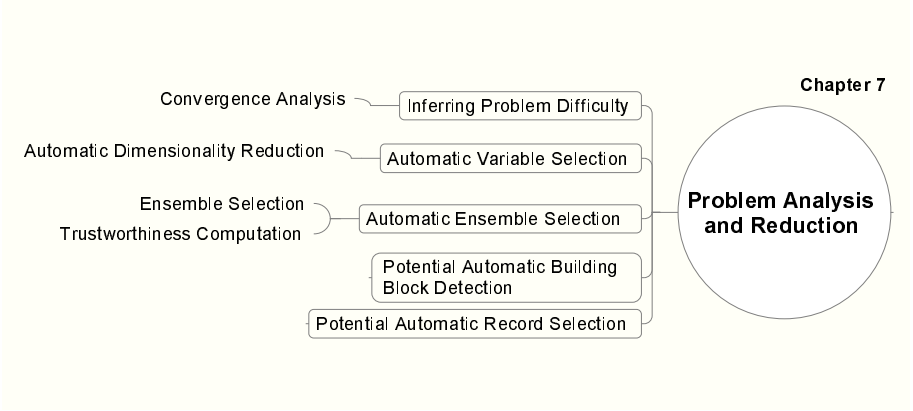


Figure 1.6: Layout of the phase of problem analysis and reduction presented in this thesis.

development stage. We, however, summarize them in a separate Part of Problem Analysis and Reduction for presentation purposes.

1.5.2 Contributions

One important feature of Pareto genetic programming, and GP in general that has to be taken into account when enhancements to the evolutionary search are sought for, is the fact that all elements of the search are deeply inter-related. The same holds for the research questions of this thesis - they are in no way mutually exclusive. Changes, introduced at the phase of model evaluation, inevitably result in changes in the phase of model selection, since the latter selects promising models with respect to their evaluated quality features. Modifications of genetic operators at the model generation phase, or modified model selection strategies all imply changes in the navigation through the search space of alternative solutions. Everything comes down to the search for such enhancements of the Pareto GP that make the path to solutions shorter by improving the balance between exploration and exploitation.

Contributions to analysis and adaptation of given data: We studied ways to balance the 'messy' high-dimensional data to make the further modeling easy. Along these lines several novel methods for data interpretation and manipulation are proposed:

1. **Four simple data weighting procedures** are introduced for computing the importance of a data point relative to its k nearest-in-the-input- space neighbors. The importance or the relative information content is defined based on different notions of closeness to k nearest-in-the-input space neighbors. The proposed weights are developed to be incorporated into the fitness function for symbolic regression, but are also useful for getting a first glimpse of insight on the relative imbalancedness of data before the modeling.

2. For enhanced analysis of large and imbalanced data sets we introduce **a Simple Multi-dimensional Iterative Technique for Sub-sampling**. The SMITS technique allows a sensible compression of multi-dimensional data to *balanced nested* subsets of arbitrary size. The balancedness is understood in relation to one of the selected weighting schemes. We use the proposed iterative procedure to **rank the records of the data set in the order of decreasing importance** and define the **cumulative information content of the resulting ranked set**. The suggested cumulative information content is used to quantify the amount of information in the compressed subsets relative to the original data set (see Chapter 2).

Contributions to reduced over-fitting: In this thesis we study the possibilities to generate regression models that not only give good predictions of the observed data but also have smoother response surfaces and better generalization capabilities with respect to extrapolation outside the observed region. Two new heuristics are presented for this purpose:

1. A new complexity measure called **the order of non-linearity** is introduced for regression models. It evaluates the behavioral complexity of the model structure, using a simplification of the notion of the degree of the best fit polynomial approximating a continuous function with certain precision. We propose two different strategies for using the order of non-linearity to construct regression models with smoother response surfaces. One way is to preform symbolic regression with a two-dimensional optimization of model accuracy and model's order of non-linearity. This

implies that the selection of potentially good models happens based on a trade-off between two competing criteria (see Chapter 4).

2. **Explicit non-linearity control** is a continuation of the work by Smits and Kotanchek (2004), who introduced explicit Pareto-aware control of expressional complexity to tree-based genetic programming. They observed that minimizing structural complexity together with prediction accuracy helps to prevent bloat and over-fitting, and focuses modeling effort on solutions which are both accurate and compact. However, since expressional simplicity does not always imply behavioral simplicity, compact models still could produce pathologies for minor extrapolations. This motivated us to combine more criteria for model selection without making the objective space unnecessary large. We proposed a heuristic for a two-objective minimization of model prediction error and model complexity, where the definition of complexity was alternated between the expressional complexity and the order of non-linearity at each iteration step. **The strategy of alternating complexity measures** allows exploiting the efficiency of the bi-objective optimization when more that two competing objectives are of interest. We show that results of experiments with alternating complexity measures are superior to those of simple bi-objective optimizations of accuracy and expressional complexity and accuracy with the order of non-linearity. We find that the new strategy applied to complexity measures produces solutions which are both compact and have smoother response surfaces, and hence they contribute better to problem interpretability and understanding (see Chapter 4).

The main contribution of this study on reduced over-fitting and non-linearity control is not the explicit definition of the order of non-linearity, which despite being generic concerns tree-based structures, and not the explicit definition of the model selection strategy for simultaneous optimization of prediction accuracy and order of non-linearity. The main contribution of this study to symbolic regression is the evidence, that structural parsimony may not imply behavioral generality, and that other kinds of parsimony must be considered for the purpose of avoiding over-fitting and producing models with good generalization capabilities.

Contributions to enhanced model evaluation strategies for obtaining

'better solutions faster': We studied the ways to improve the results of symbolic regression by changing the approach to evaluating the fitness of potential solutions in a course of a genetic programming run. Several promising directions were discovered:

1. A significant enhancement of the quality of the regression models is observed when some of the weighting functionals, introduced in item (1), are incorporated into the fitness function, and weighted regression via genetic programming is applied to an imbalanced data set. We call this approach **data balancing**, since it balances the contribution of each data record with the weight associated with the record's relative importance (see Chapter 5).

2. There are situations when large imbalanced data sets are over-sampled. Sometimes this can be guessed *a priori*, for example when the process control data comes from the measurements of the same process performed at different plant sites. We suggest to validate the guesses by constructing the cumulative information content of the data set ranked in the order of importance. This allows compressing the 'compressable' data sets to smaller subsets of a similar information content. The results of symbolic regression can be significantly enhanced, if the standard regression or weighted regression is applied to the compressed subsets of data with larger population sizes. We call this approach **exploratory regression via data balancing** (see Chapter 5).

3. The data sets for regression may contain too many variables from which only a handful are significantly related to the response. In the absence of the information about the significance of the input variables weighting data records in the original input-output space can be dangerously wrong, since the spurious variables will bias the distance-based concepts of the relative importance. Usually, before the weighting, some screening modeling runs are applied to the original data to identify driving input variables and focus the modeling effort on only those. The need to create 'better solutions faster' pushes to look for heuristics to improve the quality and robustness of regression models for the high-dimensional data sets with spurious variables. We propose two ways to do that.

(a) The first approach simply represents a new view at the old trick of using random subsets of the training data for partial fitness evaluations. The novelty (for GP) of the approach lies in a simple observation borrowed from ordinal optimization - if solutions are evaluated coarsely at a smaller computational cost, more of them can be evaluated within a fixed budget of function evaluations. We proposed to evaluate more individuals on small subsets of data drawn randomly at each generation and linearly increase with the subset size over the GP run. We called this approach the **Soft Ordinal Pareto genetic programming** and showed that it causes a significant improvement over the standard Pareto genetic programming both in the quality of solutions (on test and the training data) and in the reproducibility of the GP runs (see Chapter 6).

(b) To further improve the Soft Ordinal Pareto approach to GP, we suggested a new strategy of performing the evolutionary search. In the style of ordinal optimization we evaluate more potential solutions on smaller subsets of data, starting with 'the most important' records. This time we do not use stochastic sub-sampling of data, but the one defined by our simple multi-dimensional iterative technique for sub-sampling (see item (2) of the contributions additional to research questions), applied to the weights proposed in item (2) of the contributions additional to research questions. We use the data set ranked in the order of decreasing importance (w.r.t. a selected weight), start with a small set of the first m most important points and incrementally move the counter along the ranked records. By doing this we force the GP system to apply most effort on modeling small subsets of records carrying the essential fraction of the information content of the data set. We called this approach the **ESSENCE algorithm** since it performs an Effective Search Space Exploration via Nested Content-based Evolutions. The speed of adding new data records into nested training subsets is controlled by the shape of the cumulative information content of the ranked data. This explicitly relates the learning effort to the quality and compressibility of the data set (determined independently from the modelling tool), which has not

been presented before to the best of our knowledge[9] (see Chapter 7).

1.5.3 Overview of related papers

This dissertation includes the material presented in the following list of papers and book chapters (taken in different proportions):

Vladislavleva, E., Smits, G., and den Hertog, D. 2007. Order of non-linearity as a complexity measure for models generated by symbolic regression via Pareto genetic programming. Conditionally accepted for publication by IEEE Transactions on Evolutionary Computation.

Vladislavleva, E., Smits, G., and den Hertog, D. 2007. On the importance of data balancing for symbolic regression. *In review.*

Vladislavleva, E., Smits, G., and den Hertog, D. 2007. Symbolic regression reduced to its ESSENCE: On Ordinal Approach to Incremental Data Modeling. *In review.*

Vladislavleva, E., Smits, G., and Kotanchek, M. 2007. Better solutions faster : Soft evolution of robust regression models in Pareto genetic programming. In Genetic Programming Theory and Practice V, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation. Springer, Ann Arbor, Chapter 2, 13–32.

Smits, G. and Vladislavleva, E. 2006. Ordinal Pareto genetic programming. In Proceedings of the 2006 IEEE Congress on Evolutionary Computation, G. G. Yen, L. Wang, P. Bonissone, and S. M. Lucas, Eds. IEEE Press, Vancouver, Canada, 3114 – 3120.

Kotanchek, M., Smits, G., and Vladislavleva, E. 2006. Pursuing the Pareto paradigm tournaments, algorithm variations & ordinal optimization. In Genetic Programming Theory and Practice IV, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation, vol. 5. Springer, Ann Arbor, Chapter 12, 167–186.

[9] For corrections and constructive suggestions please contact the author at katya@vanillamodeling.com or katya@evolved-analytics.com

Smits, G., Kordon, A., Vladislavleva, K., Jordaan, E., and Kotanchek, M. 2005. Variable selection in industrial datasets using pareto genetic programming. In Genetic Programming Theory and Practice III, T. Yu, R. L. Riolo, and B. Worzel, Eds. Kluwer, Ann Arbor, MI, USA, Chapter 6, 79–92.

Kotanchek, M., Smits, G., and Vladislavleva, E. 2007. Trustable symbolic regression models: Using ensembles, interval arithmetic and Pareto fronts to develop robust and trust-aware models. In Genetic Programming Theory and Practice V, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation, vol. 6. Springer, Ann Arbor, MI, USA, Chapter 12, 203–222.

Kotanchek, M., Smits, G., and Vladislavleva, E. 2008. Exploiting trustable models via Pareto GP for targeted data collection. Accepted for Genetic Programming Theory and Practice VI, R. L. Riolo, T. Soule, and B. Worzel, Eds. Chapter 5. In Press.

Part I

Data Analysis and Adaptation

2

Data Analysis and Adaptation

This chapter considers a situation when input-output data for modeling is given (legacy data), and it is most certainly un-designed. In practice, this can be a collection of on-line and off-line physical measurements of a production process taken over various periods of time (possibly with repetition), over several plants or production machines, under various environmental conditions. Such data collections might lack any particular structure and homogeneity even in the input space. This chapter presents an approach for analysis and eventual balancing of such imbalanced input-output data sets. Presented data analysis precedes the modeling stage. It allows the modeler to automatically infer preliminary knowledge about imbalancedness and compressibility of the data set, and about relative importance of the data records.

The chapter introduces a method for automatic assignment of weights to records, that takes into account record's relative importance. The weights can be used for identification of potential outliers, concluding about the degree of imbalancedness of the data, or can be directly incorporated into the fitness function at the modeling stage. Four weighting schemes are introduced that define the importance of a point in one of the following ways: (1) proximity to k nearest neighbors in the input space, (2) surrounding by k nearest in the input space neighbors, (3) remoteness from k nearest in the input space neighbors, and (4) deviation from a least-squares hyper-plane passing through k nearest in the input space neighbors, $k \geq d$, d is the dimensionality of the input space.

For enhanced analysis and visualization of large multi dimensional data sets a new procedure for sorting the data in decreasing importance is introduced.

This procedure applied to data weighted with one of the four proposed weighting schemes permits a sensible compression of the original data to nested subsets of arbitrary size, such that these subsets are balanced with respect to the selected weight. To guide the decision about the compression rate, a notion of a cumulative information content can be defined for a data set ranked in order of decreasing importance.

The weights, ranks and cumulative information contents of imbalanced input-output data, defined in this chapter, can be incorporated into symbolic regression to significantly increase the effectiveness of the genetic programming search. The possibilities of such incorporation are presented in Chapter 3 and Chapter 5 of this thesis.

2.1 Motivation for data balancing

The need for data balancing for regression originates from applied research. Industry requires accurate prediction and optimization models, and challenges applied researchers to develop these models from imperfect data. The customers do not accept the "garbage in, garbage out" excuses and often demand robust, parsimonious, and accurate models built from a set of "more-or-less reliable observations taken under difficult conditions with possibly high inaccuracy".

The aim of any empirical modeling approach is to infer hidden dependencies from given data. For a regression problem the task is to use the given data to represent the output variables as analytical functions of *some* or all of the input variables. If the modeling approach cannot find a convincing relationship between the inputs and the outputs, it may be that the data does not contain the required information for predicting those outputs. Reasoning of this nature inspires to perform extensive analysis of the data, and data's information content *before* the modeling.

In an ideal scenario, the data for symbolic regression comprises the output observations for almost all possible combinations of inputs. We say then, that the data is *balanced*, which usually means that the data points are distributed uniformly over the input space, and consequently provide a similar amount of information about the underlying response. In this situation the modeling algorithm treats all samples equally during the learning process. In practice, however, it often turns out that certain regions of the input space

are over-represented by an abundance of observations while other regions lack representatives. When there is a difference in the information value among the data samples, we say that the data is imbalanced. Imbalanced data can come from various sources. First, it may be simply impossible to collect the data through a design of experiments, e.g. for physical measurements of a production process - you have to live with what you get. Second, even if possibilities exist to collect data in a designed fashion, merging several designs may lead to an imbalanced data set, or the design may be incomplete if some regions of the design space are infeasible and a realization of the response cannot be computed for those regions.

In general, building models on imbalanced data carries a risk of getting the solutions that perform well only on over-represented areas of the input space. This obviously impairs such important model features as robustness and generalization capabilities.

The concern about modeling with imbalanced data was first mentioned by the machine learning community in Provost (2000), but only for class imbalance problems. In classification problems, when a certain class is over-represented in the data set, the obtained solution can be biased to assign all samples to the dominant class. Of course, this is usually the opposite of what the classification model is meant to do. Therefore, in classification problems, data balancing consists in achieving an equal representation of classes with an emphasis on the boundaries between classes.

Regression can be seen as a generalized classification, where the number of classes is equal to the number of data records, and the rule describing the classes is constrained to be an analytical function of the inputs. The difficulty of finding such a rule, or a response surface, may be party caused by the fact that available points do not provide a sufficient representation of the design space in question.

It appears to the author that researchers performing the analysis of imbalanced high dimensional data are concerned primarily about outlier detection; see (Aggarwal and Yu, 2001; Harmeling et al., 2006). An underlying assumption of this chapter is that the data available for regression is reliable, but possibly with some noise. Under this condition, the 'outliers' existing in the data should be considered as important samples containing novel information about the output. The assumption about the reliability of the data motivates us to detect outliers not for a purpose of deleting them from the data but for collecting them and

treating with care during the modeling process[1].

One of the research goals of this thesis is to assess whether symbolic regression can be improved if the given data gets balanced. In essence, data balancing implies such modification of the fitness function that the performance of the generated solutions improves.

2.2 Empirical risk minimization

In symbolic regression the task is to select a function that imitates the actual system best. To accomplish this, it adjusts and combines functions from its function set until it finds a relationship that most successfully approximates the underlying dependency given a finite set of observations used as training data. To introduce the necessary notations we say, that we need to relate a set of N points in a d-dimensional input space $\mathbb{R}^d$ with a set of N points in the one-dimensional output space $\mathbb{R}$ [2] by a function $\hat{f}$: $\mathbf{x} \mapsto f(x)$, such that it approximates the output points with a smallest error.

To quantify the correctness (fitness) of the approximation we define an error- or loss function $L(y, \hat{f}(x))$. This function compares the predicted outputs $\hat{f}(\mathbf{x})$ with the observed outputs $\mathbf{y}$ and assigns a large value if the prediction is poor.

Formally, the output y is given with a fixed conditional density $p(y|x)$. For deterministic systems, this comes down to $y = \hat{f}(x)$ whereas for regression cases $y = \hat{f}(x) + \varepsilon$ with ε being a random noise with zero mean to mimic the noise in the samples. By consequence, the random term in the output of yet unseen inputs implies that the output can be represented with a probability distribution. For more details, see the book by Cherkassky and Mulier (1998).

The risk functional that describes the expected value of the loss function is

[1] This is not entirely true for real-life data. The presented techniques will order the records in importance instead of splitting them into two groups outliers - prototypes. It will be up to the modeler to decide which records have to be removed from the modeling and which should stay and be treated with care. The relative differences in values of weights computed for the most important points can guide this decision.

[2] This thesis assumes that if r-dimensional output is present, it should be modeled r times per dimension via available inputs and possibly some of the other $r-1$ output dimensions. In practice, it is often beneficial to model one dimension of the output via all inputs and the remaining $r-1$ outputs. This allows to perform sensitivity analysis of the output components, and helps to focus modeling effort on unrelated output dimensions only.

given by :

$$R(\hat{f}) = \int L(y, \hat{f}(x)) \cdot p(x, y) dx dy \tag{2.1}$$

In most real-life problems nothing is known about the underlying distribution of data. We then approximate the estimated risk (2.1) by the empirical risk, which is the average risk over the observed data $(\mathbf{x}^i, y^i)$, $i = 1, \ldots, N$:

$$R_{emp}(\hat{f}) = \frac{1}{N} \sum_{i=1}^{N} L(y^i, f(x^i)). \tag{2.2}$$

Often, in regression cases, the squared error (L2) is used as a loss function: $L(y, \hat{f}(x)) = (y - \hat{f}(x))^2$. The risk functional (2.1) becomes:

$$R(\hat{f}) = \int (y - \hat{f}(x))^2 \cdot p(x, y) dx dy, \tag{2.3}$$

and is estimated by the averaged sum of squared errors on a finite set of training samples:

$$R_{emp}(\hat{f}) = \sum_{i=1}^{N} \frac{1}{N} (y^i - \hat{f}(\mathbf{x}^i))^2. \tag{2.4}$$

The definitions of these theoretical and empirical risk functions imply that the areas of the data space with fewer observations will not have a large influence on the learning process since they are "averaged out" in the equation (2.2), or 'weighted out' by small probabilities in the equation (2.1). By consequence, the final prediction model will be inclined to perform well *only* on the most common data points (further referred to as *prototypes*).

We aspire to overcome this problem by incorporating the "information weight" of a data point into the modeling process and adjust the empirical functional (2.4) in the following manner:

$$R_{emp}(f, w) = \sum_{i=1}^{N} w_i (y_i - f(x_i))^2 / \sum_{i=1}^{N} w_i. \tag{2.5}$$

With the modified risk functional, models will be discouraged to have inaccuracies in the points with higher weights. Since more important data points will have more influence on the learning process, we expect that such data

balancing will improve the performance of resulting models not only in terms of the weighted accuracy and inter-and extrapolation capabilities, but also in terms of the absolute accuracy on the test data[3] .

We illustrate in Figure 2.1 that using weights is in fact a more general way of approximating the theoretical risk functional (2.3). When the joint probability density function of the data $p(x, y)$ is not known, the weights may work as the approximations of the sample density while incorporating some other features of the data samples (see section 2.5).

Knowing, that by adjusting the weights, we can control the influence of each data point on the computation of the empirical risk functional, and also modify the risk functional in such a way that it can be minimized with less effort, we need to find proper ways for weight assignment. Formulae for the theoretical and empirical risk functionals suggest that the weights in the weighted risk functional have to be distance based, relative, and preferably reflect the local density of the data in the neighborhood of a point. The other aspects related to the weight generation are presented in the following section.

2.3 Essential aspects of data balancing

Appropriate distance metrics

When looking for appropriate and scalable distance-based weighting schemes for multi-dimensional data we should start with having a scalable distance metric. The curse of dimensionality hinders a Euclidean distance metric $L_2(p, q) = \left(\sum_{j=1}^{d}(p_i - q_i)^2\right)^{1/2}$, $p, q \in \mathbf{R}^d$ from giving a meaningful notion of proximity in a high-dimensional space. We suggest using a fractional distance metric $L_{1/d}$ in a d-dimensional space, when d is large:

$$dist_{1/d}(\mathbf{p}, \mathbf{q}) = \left(\sum_{i=1}^{d} |p_i - q_i|^{1/d}\right)^{d},$$

[3]The ambitious goal of the experiments of this chapter is to demonstrate that symbolic regression with minimizing the weighted error lead to solutions that are superior to the results of the unweighted error minimization with respect to the **unweighted error on the test data**.

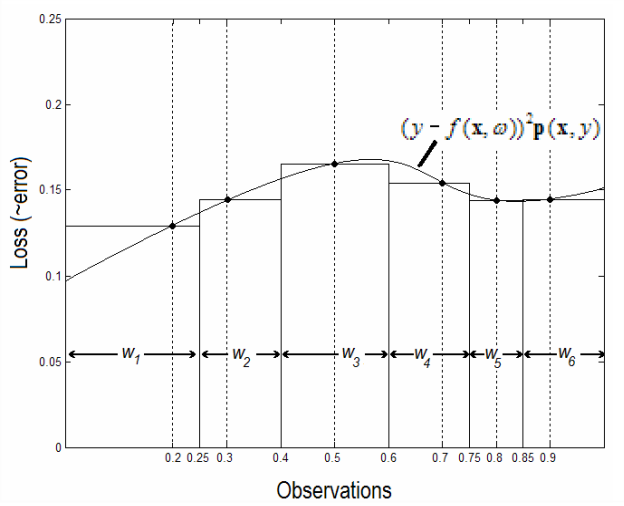


Figure 2.1: General approximation of a theoretical risk functional by using weights.

$$L_{1/d} : \|\mathbf{q}\|_{1/d} = \left(\sum_{i=1}^{d} |q_i|^{1/d} \right)^d .$$

The compelling evidence of the better scalability of a fractional distance metric in a space of high dimensionality can be found in (Aggarwal et al., 2001), where the fractional distance metrics $L_{1/z}$ are introduced for an arbitrary $z \in \mathbb{Z}$. In (François et al., 2007) the relevance of using fractional distances with respect to the distance concentration phenomenon is discussed in detail.

Appropriate input dimensions

Another pitfall of manipulating the multi-dimensional input-output data is the initial absence of any information about the significance of the contribution of inputs into the underlying input-output relationship. The reader should be warned about the necessity to perform the data balancing in the subspace of data containing only significant inputs. Some techniques for automatic variable selection in symbolic regression via genetic programming are described

in (Kotanchek et al., 2007; Smits et al., 2005) and in Chapter 8[4].

When the significant inputs are identified, it is useful to scale all the variables to the same range, to ensure that no unadapted data ranges bias the data weights and the interpretation of the data[5].

Appropriate meaning of the information content

According to intuition, the following three criteria should constitute the *relative* importance of a data point in a given input-output data set:

1. **Proximity to the neighbors.** Isolated points with *novel* information are easy to ignore in the global response surface modeling. Emphasizing these points with higher weights may improve the accuracy of solution in the under-represented regions.

2. **Surrounding by the neighbors.** Points that are not uniformly surrounded by neighbors may be seen as the edge points of the data. Detecting and emphasizing the edge points may improve the accuracy of extrapolation into the 'unseen' areas of the input space.

3. **The shape of the underlying response surface in the neighborhood of the point.** Emphasizing the areas of non-linear changes in the response may improve the accuracy of solutions. Compressing the data to only such areas will allow spending more computational budget, i.e. more modeling efforts, on 'interesting' areas of the response surface, and may increase the probability of finding better solutions within the same computational time.

2.4 Existing methods and useful concepts

2.4.1 Voronoi diagrams

As we stated in the previous section, one of the aspects of the importance of a data point may be seen as the proximity to the neighbors of the point, or in other words, the amount of 'space' around the point, for which this point is responsible.

[4]In some applications the meaningful input variables are not equally relevant for the process in question. In these cases we suggest using relative 'weights' for the dimensions in the definition of a distance metric.

[5]The rule of thumb in "To scale, or not to scale" question is to always scale the input variables, if they are equally important for the modeler.

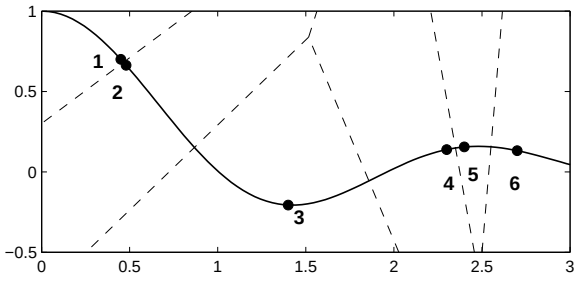


Figure 2.2: Size of a Voronoi polygon as a value for the information content.

Consider an example with six points sampled from a response curve plotted in Figure 2.2. We may speculate that the 'information content' of point 3 is higher than the one of point 4 or of point 5, since point 3 solely carries the responsibility for the central part of the input space X and the input-output space XY.

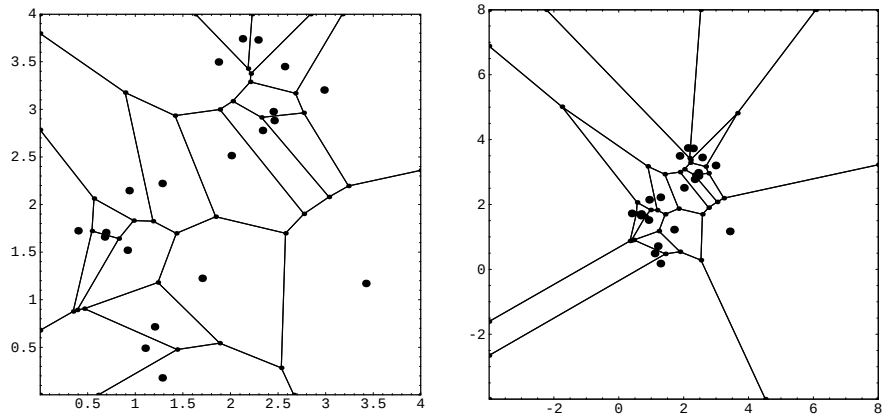
Since the amount of 'empty space' around a data point can be quantified as a volume of a Voronoi polytope from a Voronoi diagram defined by the data set, Voronoi diagrams provide a good intuitive basis for determining weights of the data and a good approximation of the risk functional 2.1[6].

Figure 2.2 illustrates that an isolated or an edge point generates a larger Voronoi cell than a point from a dense region in both input space and the input-output space. If we assume that an isolated point 'represents' a larger area of the data space, and hence carries a larger information content *relative* to the points in the dense clusters, the volumes of Voronoi polygons become an intuitively comforting definition for weights of data points.

However, some reservations are restraining from defining the weights as volumes of corresponding polytopes in a Voronoi diagram generated by the data set. These reservations are the ambiguity and the computational complexity of Voronoi diagram calculations.

An unbounded Voronoi has an infinite volume and needs to be constrained by a bounding box. On one hand, a big bounding box around the data will make the edge points have big Voronoi cells and may provide a way to detect the edges of the data. On the other hand, the freedom to choose the location

[6]A Voronoi diagram can be described as the partitioning of a plane with n points into convex polyhedrons such that each polyhedron contains exactly one generating point and every point in a given polyhedron is closer to its generating point than to any other, see Okabe et al. (2000)



(a) Voronoi diagram with a small bounding box

(b) Voronoi diagram with a big bounding box

Figure 2.3: Relative differences in volumes of Voronoi polytopes change with a size of a bounding box.

and the size of a bounding box makes the definition of Voronoi cells, and hence, the relative weights of points ambiguous; see the examples in Figure 2.3. In higher dimensions the problem of ambiguity becomes more severe, since the size of Voronoi cells corresponding to the edge points grows a lot faster than the size of Voronoi cells, corresponding to the inner points.

Although Voronoi cells are simple polygons in the two-dimensional case, they become complex polytopes in high-dimensional situations. This is the reason why calculating the volume of Voronoi-cells becomes computationally hard when more points and/or dimensions are taken into account. The main reason of hardness of generating a Voronoi diagram is a problem of finding a convex hull of N points in a d-dimensional space. The computational complexity of theoretically optimal algorithms is $O(N^{\lfloor (d+1)/2 \rfloor} N \log N)$ for an incremental algorithm, optimal for even-dimensional problem, and $O(N^{\lfloor d/2 \rfloor} + N \log N)$ for a divide-and-conquer method, optimal for an odd-dimensional problem, see Okabe et al. (2000). This implies that for high dimensional cases, one always needs to rely on numerically stable approximations and dimension-specific algorithms to compute Voronoi cells and their respective volumes.

The summary is that the volume of the Voronoi cell corresponding to a data point is inapplicable for the weight definition in high dimensional problems, while

being intuitively appealing in low dimensions.

2.4.2 Other useful concepts

The concluding hypothesis of the previous subsection is that the weights based on the volumes of Voronoi cells do not improve the solutions of the low-dimensional problems due to the lack of relative information about the neighborhood of a Voronoi cell. A literature search for useful proximity-based concepts resulted in a discovery of a paper of Harmeling et al. (2006), with an inspiring approach to data analysis for outlier detection.

Below we introduce notations for data records used throughout this thesis. By an input-output data set we mean a set $\mathcal{M} = \{M_1, \ldots, M_N\}$ of N points in a $(d+1)$-dimensional space $\mathbb{R}^{d+1}$. Point M_i has coordinates $(x_1^i, x_2^i, \ldots, x_d^i, y^i) \in \mathbb{R}^{d+1}$, $i = \overline{1, N}$, with y^i corresponding to the response value at the input point $P_i = (x_1^i, x_2^i, \ldots, x_d^i) \in X \subset \mathbb{R}^d$. We say that the input-output point M_i represents the input point P_i, since the projection of M_i on the input space $X \subset \mathbb{R}^d$ is exactly P_i. The set of all input points is denoted as $\mathcal{P}$, the vector of outputs as $Y = (y^1, \ldots, y^N)^T$.

By $\{n_1(P_i, \mathcal{P}), n_2(P_i, \mathcal{P}), \ldots, n_k(P_i, \mathcal{P})\} \in \mathcal{P}$ we denote the k nearest neighbors of the point $P_i \in \mathcal{P}$ in metric L_2 or $L_{1/d}$.

For each point $P_i \in \mathcal{P}$ the following functions will be used as a basis for weighting (adopted from Harmeling et al. (2006)).

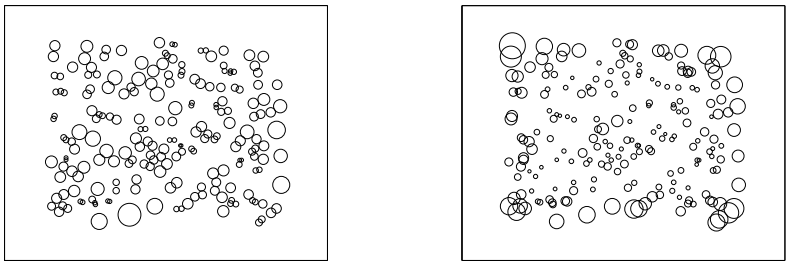
(1) **Weight based on the proximity of the k nearest neighbors in the input space X** defined as the average distance to the k nearest neighbors (**proximity-X**):

$$\pi(i, \mathcal{P}, k) = \frac{1}{k} \sum_{j=1}^{k} \|P_i - n_j(P_i, \mathcal{P})\|, \tag{2.6}$$

where $n_j(P_i, \mathcal{P})$ is the j-th nearest neighbor of the point P_i from the set $\mathcal{P}$ in the norm $\|\cdot\|_{1/d}$ or $\|\cdot\|_2$.

(2) **Weight based on the surrounding by the k nearest neighbors**, defined as the length of the average of the vectors pointing at the k nearest neighbors (**surrounding-X**):

$$\sigma(i, \mathcal{P}, k) = \left\| \frac{1}{k} \sum_{j=1}^{k} (P_i - n_j(P_i, \mathcal{P})) \right\|, \tag{2.7}$$



(a) Proximity, $k = 1$ (b) Surrounding, $k = 20$

Figure 2.4: Detecting isolated and edge points with proximity-X (2.6) and surrounding-X (2.7) indexes. Radii of the circles correspond to the relative values of proximity-X index in the left plot and the surrounding-X index in the right plot.

where $n_j(P_i, \mathcal{P})$ is the j-th nearest neighbor of the point P_i from the set $\mathcal{P}$ in the norm $\|\cdot\|_{1/d}$ or $\|\cdot\|_2$.

The neighborhood size k dictates the scale in the perception of the data - small neighborhood size suggests local analysis of data, while big neighborhood size implies a global view on data. Isolated points are detected best via the proximity index with the neighborhood size of one, and edges of data are detected at best with a large neighborhood size, e.g. $N - 1$. We illustrate this in an example in Figure 2.4. Figure 2.4 presents a set of 200 points is randomly sampled from a two-dimensional interval $[0, 0.5] \times [0, 0.5]$ and weighted according to the proximity-X index 2.6 with the neighborhood size of one (plot (a)), and the surrounding-X index 2.7 with the neighborhood size of 20 (plot(b)). Radii of the circles correspond to the relative values of proximity-X index in the left plot and the surrounding-X index in the right plot.

2.5 New definitions for weighting functionals

2.5.1 Including the response

In real-life applications the input data does not necessarily come from a designed experiment, may contain some noise, or may not be uniformly distributed if some regions of the design space are infeasible due to the physical limitations of the

experiments or computational limitations of the simulation system. These cases raise a question: - when the "messy" data is given, in which space do we need to balance it, in the input space or in the total input-output space?

When we have freedom to design a limited set of samples for modeling a non-linear response surface at a certain hyper-interval of the input space, we want the sampled points to form a balanced 'space-filling' subset of that hyper-interval. This is a motivation for the design of experiments, where the design points are chosen to form a maximally 'regular', space-filling mesh of the design space.

When we model given data, our task is to approximate the underlying response surface with a maximal precision. For this, we would prefer our points to form a finer mesh in the areas of high non-linearity of the response surface and coarse mesh in the areas where the response surface is linear.

2.5.2 Proximity and surrounding weights

Motivated to include the information about the outputs into the data-balancing procedure we need to adjust the definitions of the proximity-X and surrounding-X weighting schemes.

Note that the proximity-X and surrounding-X weights of points in the input space with respect to k closest neighbors can be plausibly determined with the functionals (2.6) and (2.7), when the neighborhood size k is fixed. In the input-output space, however, the notion of closeness to the closest neighbors can be deceiving. If a set of points is located on a d-dimensional manifold defined by the response surface, then the points closest to each other in the $(d+1)$-dimensional input-output space, are not necessarily the closest on the manifold. Such situation is illustrated in Figure 2.5.

We suggest to use this simple observation in adapting the proximity and surrounding weights for response surface modeling.

We define the new **weight based on the proximity by k nearest-in-the-input-space neighbors** (**proximity weight**) as:

$$\pi(i, \mathcal{M}, \mathcal{P}, k) = \frac{1}{k} \sum_{j=1}^{k} \|M_i - \bar{n}_j(M_i, \mathcal{M}, \mathcal{P})\|, \tag{2.8}$$

where $\bar{n}_j(M_i, \mathcal{M}, \mathcal{P})$ is the j-th nearest-in-the-input-space neighbor of point M_i in the set $\mathcal{M}$. This means that the projection of $\bar{n}_j(M_i, \mathcal{M}, \mathcal{P})$ onto the input

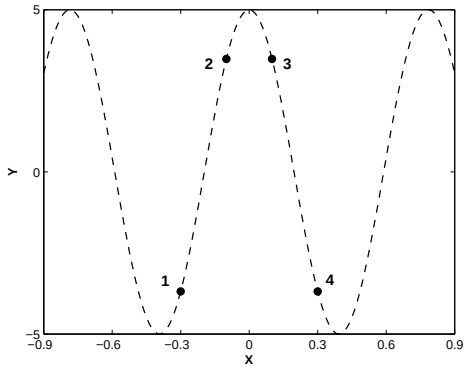


Figure 2.5: Proximity to the closest neighbors in the input-output space may be deceiving. According to this plot, the closest neighbor of point 1 in XY-space is point 4. However, along the response curve point 1 is the furthest apart from point 4.

space X is $n_j(P_i, \mathcal{P})$.

We use the same idea of determining the neighborhood in the input space in the new definition of the **weight based on the surrounding by k nearest-in-the-input-space neighbors** (**surrounding weight**):

$$\sigma(i, \mathcal{M}, \mathcal{P}, k) = \left\| \frac{1}{k} \sum_{j=1}^{k} (M_i - \bar{n}_j(M_i, \mathcal{M}, \mathcal{P}) \right\|, \tag{2.9}$$

where $\bar{n}_j(M_i, \mathcal{M}, \mathcal{P})$ is the j-th nearest-in-the-input-space neighbor of point M_i in the set $\mathcal{M}$. This means that the projection of $\bar{n}_j(M_i, \mathcal{M}, \mathcal{P})$ onto the input space X is $n_j(P_i, \mathcal{P})$.

Coming back to Figure 2.5, we observe that according to the new definitions, the proximity and surrounding weights of point 1 will be equal to the length of the vector connecting points 1 and 2, since the nearest-in-the-input-space neighbor of point 1 is point 2.

In formulae (2.8) and (2.9) a Euclidean distance metric can be used if the dimensionality d of the input space is low. However, if d is high, the fractional distance metric $\| \cdot \|_{1/(d+1)}$ should be used instead of the Euclidean, since it produces considerably better results (see, e.g., (Aggarwal et al., 2001)).

2.5.3 Remoteness weight

The features of the proximity and the surrounding weights introduced by Harmeling et al. (2006), and adjusted for the input-output data in the previous section, match our requirements for a good data weighting procedure. We want to assign higher weights to the points with a higher information content *relative to the neighbors*. High weights of these points plugged into the fitness function will drive the system to produce lower errors in specified locations. Therefore, we can assign higher weights to the edge points to constrain solutions to exhibit more accurate extrapolation behavior, and assign higher weights to the points in the sparse regions - to force solutions to agree with the training data in under-represented areas that are easy to ignore otherwise.

The goal of this section is to understand whether the definitions of proximity and surrounding weights (2.6) and (2.7) are sufficient for detecting the edges of data as well as the regions of sparsity. A high proximity index detects points in the sparse areas, while a high surrounding index detects the points that are not surrounded uniformly, i.e. are located on the 'edge'.

Harmeling *et al.* use indices implied by (2.6) and (2.7) primarily for detecting outliers. For these purposes the surrounding index is preferable, since it identifies points that are both isolated and not surrounded uniformly by their neighbors. Using the index implied by the surrounding weight, the authors successfully divide the data set into two groups - outliers and prototypes, prune away the outliers, and exploit a representative set of points sharing 'common' features.

Our purpose of weighting is opposite to some extent. Having agreed that the incoming data is reliable (however, possibly with noise), we want a *quick* procedure that identifies points with 'untypical' features from the rest, and assigns higher weights to these points. We do need 'outliers'. We do want to leave a *representative* set of points exhibiting novelties in data and prune away the redundancies to ease and speed up the regression process. That is why we want to assign high weights to the points with either a high proximity index or a high surrounding index.

Taking the average of the proximity and surrounding values does not produce a satisfying combination, since the very high values of one weight function can be decreased by the very low values of another weight function.

A multitude of ways was tried to combine the proximity and surrounding

indexes into one combined weight. The method that is intuitively comforting and also gives the best results on various problems is ordinal, since it combines the ranking of data points, not the actual weight values.

Let $w : \mathcal{M} \mapsto w(\mathcal{M}, \mathcal{P}, k) \in \mathbb{R}$ be a weight functional of the set of points $\mathcal{M} = \{M_1, M_2, \ldots, M_N\}$ in $\mathbb{R}^{d+1}$. The weight vector $\big(w(1, \mathcal{M}, \mathcal{P}, k), \ldots, w(N, \mathcal{M}, \mathcal{P}, k)\big)$ induces a certain ranking of values w. Let $\mathcal{I}[i; w(\mathcal{M}, \mathcal{P}, k)] \in \{1, \ldots, N\}$ be an index of the value $w(i, \mathcal{M}, \mathcal{P}, k)$ in that ranking.

The **weight based on the remoteness of the point M_i from k nearest-in-the-input-space neighbors** is defined as the *rank* of the average of its proximity and the surrounding ranks:

$$\rho(i, \mathcal{M}, k) = \mathcal{I}\left[i; \mathcal{I}\left[i; \pi\left(i, \mathcal{M}, \mathcal{P}, k\right)\right] + \mathcal{I}\left[i; \sigma\left(i, \mathcal{M}, \mathcal{P}, k\right)\right]\right]. \tag{2.10}$$

This definition means that instead of taking the average of the values of the proximity (2.8) and the surrounding(2.9) weights for a given point, we compute the proximity and surrounding -Y weights for all points in the data set, for each point we find a rank induced by the two weighting schemes, take the sum of the ranks, rank these sums again, and define the remoteness weight of the point in question as an index of this point in the final rank. By definition, the value of the remoteness weight $\rho(\cdot)$ is an integer value from the set $\{1, 2, \ldots, N\}$. We can also normalize the remoteness weights to sum up to the number of points N. In this case the remoteness weights get a minimum value of $2/(N + 1)$, and the maximum value of $2N/(N + 1)$.

The presented way to combine weights is normalization-invariant, and is also more robust to the possible noise in the data. According to the new definition, irrespectively of the degree of imbalancedness of the data, only one half of the data gets weights higher than one. Besides, the highest remoteness weight is only N times higher than the lowest remoteness weight. Therefore, the isolated points with very high proximity or surrounding values cannot get excessively large weights and create an artificial bias for regression.

For completeness, the combined remoteness weight needs to be determined for the input space. Let us call it the **weight based on the remoteness from k nearest neighbors** (**remoteness-X**), and define it as a combination of ranks induced by π and σ weights:

$$\rho(i, \mathcal{P}, k) = \mathcal{I}[i; \mathcal{I}[i; \pi(i, \mathcal{P}, k)] + \mathcal{I}[i; \sigma(i, \mathcal{P}, k)]]. \tag{2.11}$$

2.5.4 Non-linearity weight

Now we want to analyze and define the capabilities and limitations of the remoteness index. As stated in the previous section, one of the additional motivations to weight the data (besides improving making the modeling easy by using a weighted fitness function) lies in separating the data into important points and redundant points, to further focus the modeling effort on the important points only. This is the reason why we like the ordinal rank-based approach - a 'sensible' ordering of data allows us to make a plausible separation of the data into two groups, while not being too sensitive to the actual weighting function.

On one hand, the remoteness index $\rho(\cdot)$ in Eq.(2.10) can successfully detect points of high variation in the response relative to the nearest neighbors. Emphasizing the points of big changes in the response with the remoteness index can be a way to capture the areas of steepness of the response surface - a non-obvious task in high-dimensional spaces.

On the other hand, over-focusing on points of high variation in the output may withdraw our attention from the areas of lower but still highly *non-linear variation*. Steep, but linear areas of the response surface are neither challenging nor significant from the point of view of non-linear modeling or extrapolation, while the areas of highly non-linear behavior may be symptoms for generating pathologies in case of extrapolation. This observation brings us to a new definition of a weighting function, that focuses not on the actual local variation of the output, but on the *local non-linearity*.

The **weight based on the local deviation from linearity** in the point $M_i \in \mathbb{R}$ is defined as the distance from point M_0 to the least-square hyper-plane approximating k nearest-in-the-input-space neighbors, $k \geq d + 1$:

$$\nu(i, \mathcal{M}, \mathcal{P}, k) = dist_{XY}(M_i, \Pi_i), \tag{2.12}$$

where Π_i is a $(k-1)$-dimensional hyper-plane passing through k nearest-in-the-input-space neighbors of M_i, and $dist_{XY}$ is the selected distance metric of the input-output space.

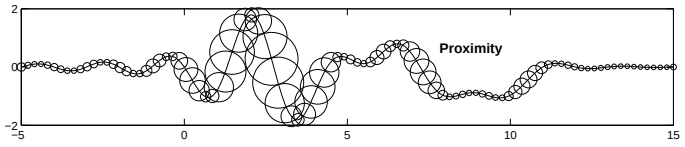
In Figure 2.6 the differences in the proximity, the surrounding, and the non-linearity weights for a one dimensional problem are plotted. The **Ksinc** function is defined by

$$y = \sum_{i=1}^{10} sinc(x - i)(p_i^2 - 1), \tag{2.13}$$

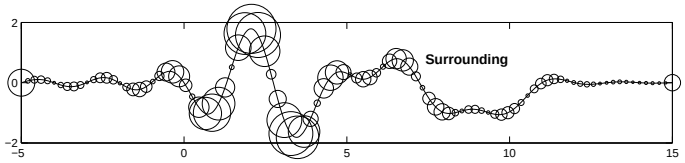
where $sinc(x) = \sin(\pi x)/(\pi x)$, p is a vector of 10 pseudorandom numbers drawn from a normal distribution with zero mean and the standard deviation one, generated by MatLab after the random number generator was set to the initial default state, i.e. "randn('state',0); p=randn(10,1);". According to Figure 2.6, the non-linearity weights for the Ksinc data are capturing all the humps of the underlying response curve the best. We speculate here, that the non-linearity weights may provide a good basis for compressing the data, since they emphasize a small number of points with higher weights around the humps, i.e. in the areas of high local non-linearity, and disregard the points, that can be obtained by a piece-wise linear interpolations through the points with the high weights. If this is true, due to generality of the definition of the non-linearity weights, the same should hold for higher dimensions.

If a data point lies on the plane going through the d nearest-in-the-input-space neighbors, the point does not bring any novel information about the output, relative to the information brought by the neighbors. The appeal of the non-linearity weight is in the fact that it focuses on the areas of the data space where the *local* change in the underlying response surface is not trivial (i.e. not linear). When modeling real-life processes driven by the fundamental laws of nature, we expect the underlying response surfaces to be relatively smooth functions exhibiting a *reasonable* behavior. This means that the expected number of the areas of highly non-linear behavior is strongly smaller than the total number of available data points[7].

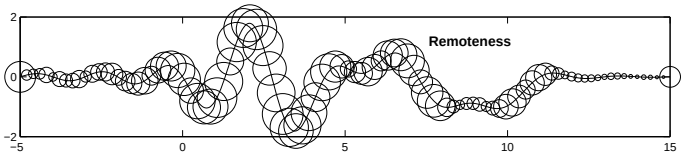
[7]The non-linearity weight should be used only if $d << N$, i.e. the dimensionality of the input space is strongly smaller that the number of records. By definition, as well as due to the area of application, the non-linearity weights are not applicable to regression on fat arrays, where $d >> N$. The latter problems, heavily under-sampled in the data space, require a dual weighting of the inputs variables, variable sensitivity analysis, and further selection of significant inputs (also known as variable selection), see Smits et al. (2005).



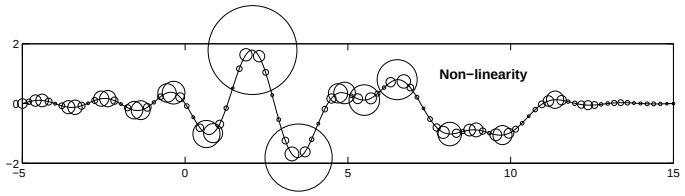
(a) Proximity Weights



(b) Surrounding Weights



(c) Remoteness Weights



(d) Non-linearity Weights

Figure 2.6: Comparison of proximity, surrounding, remoteness and non-linearity weights for a one dimensional Ksinc problem. The plots represent relative weights of 100 points sampled from **Ksinc** function. Please observe, that the non-linearity weights are capturing all the humps of the underlying response surface the best. The linear fragments of the response curve indeed have the low non-linearity weights. We speculate, that the non-linearity weights may provide a good basis for compressing the data, since they emphasize a (relatively) small number of points with higher weights around the humps, and disregard the points, that can be obtained by a piece-wise linear interpolation through the points with the high weights.

2.5.5 Computational complexity

When a neighborhood size k is fixed, the vectors of proximity and surrounding indexes can be computed in $O(N^2 d + N^2 \max(k, \log N))$ arithmetic operations, see (Harmeling et al., 2006).

For the non-linearity weight calculation, determining the plane approximating k nearest-in-the-input-space neighbors, requires solving a system of k linear equations. The complexity of the non-linearity calculation becomes $O(Nk^3)$ [8].

2.5.6 Summary of new weighting methods

In this section we defined four weight functionals emphasizing different aspects of the relative importance of a data point. All four functionals use the input-output coordinates of a point and its k nearest-in-the-input-space neighbors. We suggest to use the neighborhood size of one or $d+1$ for the proximity, surrounding and remoteness weights, and at least $d+1$ for the non-linearity weights. In the non-linearity weight definition it is possible that the plane passing through the k nearest-in-the-input-space neighbors is ill-defined. In those cases we suggest to increase the neighborhood size and find the hyper-plane approximating the k neighbors in the least-square sense.

The goal of the presented research is twofold - we want to find out whether the weighted regression produces results superior to the ones produced by the standard regression, and also find robust ways to consistently improve regression results for real-life data. Our hypothesis is that the latter can be achieved if *more* modeling effort is applied to a *smaller* subset of data with a similar information content. In the next section we introduce a novel technique for data ranking that allows adaptive selection of data subsets that contain the 'essential information' about the underlying response surface.

2.6 Using weighting for compression

2.6.1 SMITS procedure

Every weighting scheme, introduced in the previous section, can be used to rank the data points from the least 'important' ones to the most important ones. The

[8] For all weighting schemes only significant variables should be considered as an input space.

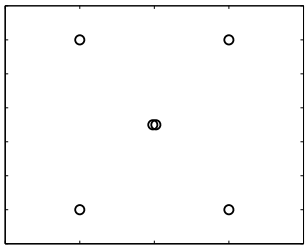
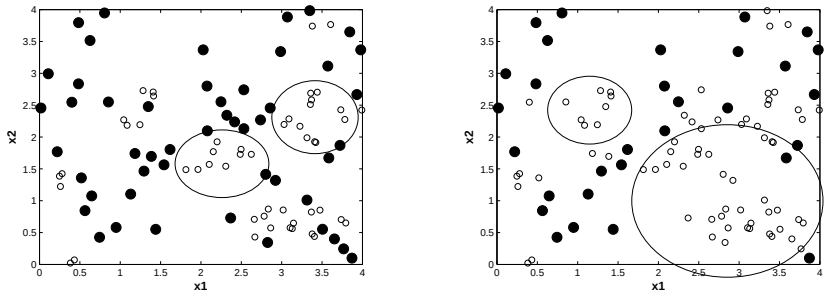


Figure 2.7: One pass weighting of a point in a dense cluster. Figure represents six points, two of which form a dense cluster that is remote from the other four points. If the neighborhood size of *one* is used for weight definition, the two close points will get very low values for the proximity, surrounding, and remoteness weights. In this case, a weight-based compression used improperly may delete both points due to their low weight values. However, collectively, the points in the cluster are representing a big central area of the data space. Intuitively, the presented data set can be compressed to five points only, if the cluster is substituted by one point. Removing more points will result in a considerable information loss.

trivial way to compress the data set would be to chop off a certain percentage of points with lowest weights in that rank. However, this would not be a smart way to compress, due to the relative nature of our weight definitions.

As we stated previously, the high values of the proximity, surrounding, remoteness, and the non-linearity weights will detect the points located in sparse, not-surrounded, or remote areas of the data space or in the areas corresponding to the high non-linearity of the response surface. The opposite statement about the points with the low weight is not true, i.e. the points with very low weights may be located close to their nearest neighbors (or be in a linear relationship with them), but be very remote from (or very non-linearly related to) the data as a whole. So, a data point gets a low weight, when it is located in a small dense cluster, which size is a bit bigger than the neighborhood size used in the weight definition, see an example in Figure 2.7.

Another example of an improper compression based on the rank generated by a remoteness index for a two-dimensional data set is given in Figure 2.8. After weighting the 100 points of the selected data set with a remoteness index, defined in equation (2.11), we select 50 and 30 points with the highest remoteness weights and plot them in Figures 2.8(a) and 2.8(b). The plots illustrate that the compression based on the one-pass calculation of the remoteness weight may miss the dense regions of the data space.



(a) Compression to 50 points (b) Compression to 30 points

Figure 2.8: Compressing a data set of 100 **imbalanced points in the INPUT space using remotness-X weights**. This imbalanced two-dimensional data set will be further considered as input set for KotanchekImbalanced test problem (see Chapters 3 and 5). Both plots illustrate that the compression of these two-dimensional data with the remoteness-X weight is not a convincing way of selecting 'space-filling' subsets of the data. The plots represent the 'top' 50 points and the 'top' 30 points with the highest remoteness weights obtained by the formula (2.11) with the neighborhood size of two. We see that points with high remoteness values (50 and 30 points with the highest remoteness values are depicted as black circles) are not remote from their two nearest neighbors but are not necessarily uniformly remote from each other.

Since the points with low weight values (according to a selected importance criterion) have a low importance only *relative* to their k nearest or k nearest-in-the-input-space neighbors, the rank generated by the selected weighting cannot be used for compression and should be adapted to reflect the importance of points in relation to the entire data set. We suggest to do it via the following **Simple Multi-dimensional Iterative Technique for Sub-sampling** (SMITS). The technique iteratively removes points with the smallest weight from the data set. At each iteration step, the weights of all points that contain the eliminated point among k nearest or nearest-in-the-input-space neighbors get re-evaluated. The order of elimination generates a ranking of data of decreasing importance, such that the most important points are eliminated last.

Let $w : \mathcal{M} \in \mathbb{R}^{d+1} \mapsto \mathbb{R}$ be a weighting functional that assigns weight to a selected set of points $\mathcal{M}$ using the neighborhood size of k. The SMITS procedure assigns rank to the data samples in the following way:

1. Weight the entire data set with w;

2. Find the point with the smallest weight;

3. Remove the selected point from the data set and remember the iteration step;
4. Update the weights of points that had the eliminated point among k nearest neighbors;
5. Continue steps(2)-(4) until k points are left in the data set.
6. Rank the points by the order of elimination, and randomly assign ranks from $N-k+1$ to N to the last k points.

The algorithm is based on iterative elimination of points with the smallest obtained weight from the data set, and ranking these points by the order, in which they are eliminated. Since all presented weighting procedures exploit the notion of k nearest or nearest-in-the-input-space neighbors, only the weights of those neighbors of the eliminated point must be updated during one iteration step. At the end of the elimination procedure only k remain in the data set, and those points are randomly ranked by indexes $N-k+1,\ldots,N$. The elimination rank corresponds to the global relative importance of a data point. *If the data is compressible* then the points with low elimination ranks are the prototypes. As many of them can be removed (with caution) as the user desires.

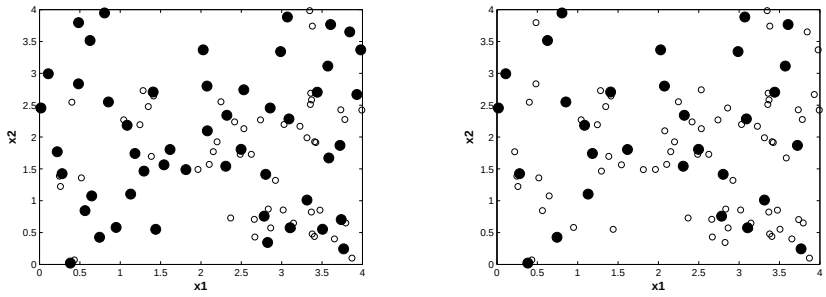
2.6.2 Space-filling compression in the input space

The elimination rank obtained by the iterative pruning of the least important points with distance-based weights (proximity and remoteness) allows a very efficient building of space filling subsets of data in the input space. The beauty of the SMITS procedure lies in the fact that with the elimination rank we can generate subsets of data which are *nested*, and, for each subset size, are the 'most' space-filling with respect to the selected weight functional.

The imbalanced data set from Figure 2.8 can now be convincingly compressed with the SMITS procedure to nested space-filling sub-sets of various size, see Figure 2.9.

In Figure 2.10 we introduce two two-dimensional data sets, **SUITS** data and **BAGEL** data, to illustrate the capabilities of the introduced iterative technique in data compression with the remoteness weight.

Figure 2.11 illustrates the SMITS-based compression of the **SUITS** data set with the remoteness weight. We used three different neighborhood sizes to

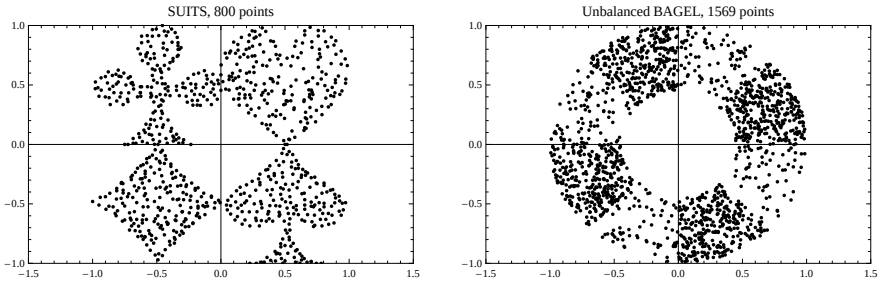


(a) Compression to 50 points (b) Compression to 30 points

Figure 2.9: Compressing a data set of 100 **imbalanced points in the INPUT space via the SMITS procedure with remoteness-X weights**. Both plots illustrate that the subsets of the first 50 and 30 points obtained from the ranking generated by the SMITS procedure applied to the remoteness-X weights are sufficiently space-filling in the given x_1x_2-space. Points selected into the subsets, are depicted as filled circles. Note that the 'space-filling' set of 30 points in plot (b) is the subset of 50 points depicted in plot (a). Iterative elimination of points with the smallest remoteness-X weights allows a good compression of dense areas. This is the reason, why problems with the compression based on the simple sorting of the remoteness-X weights (see Figure 2.8) disappear with the SMITS-based compression.

demonstrate the flexibility of the iterative elimination principle with respect to the user goals. We believe that the neighborhood size of one is a robust and sufficient setting for creating the space-filling subsets with the proximity and the remoteness weights. When compression is performed for edge detection, the neighborhood size should be increased. The choice for k will depend on the original size of the data set and also on the desired compression rate. We illustrate this property using compression for emphasis and detection with the neighborhood sized of one and five percent of the original data size.

The introduced iterative procedure does not suffer from the flaw of missing the dense regions compared with the compression based on the one-pass weight calculation. If the data has a dense cluster with a size bigger than the considered neighborhood size, all points in this cluster get small weights (even if the non-linearity weight is calculated, the densely sampled regions of the response surface may appear locally linear, and may obtain small weights relative to the points outside the dense cluster). Compression via the iterative procedure will eliminate all but one point from the cluster. That last point will be further eliminated only if all other points outside the cluster are more dispersed from each other than



(a) Card Suits Data Set (b) Unbalanced Bagel Data Set

Figure 2.10: Examples of data sets in the input space. To illustrate the capabilities of the SMITS procedure for creating space filling subsets of data, we generated two test sets with a non-trivial edge structure, and non-uniform density in the data space.

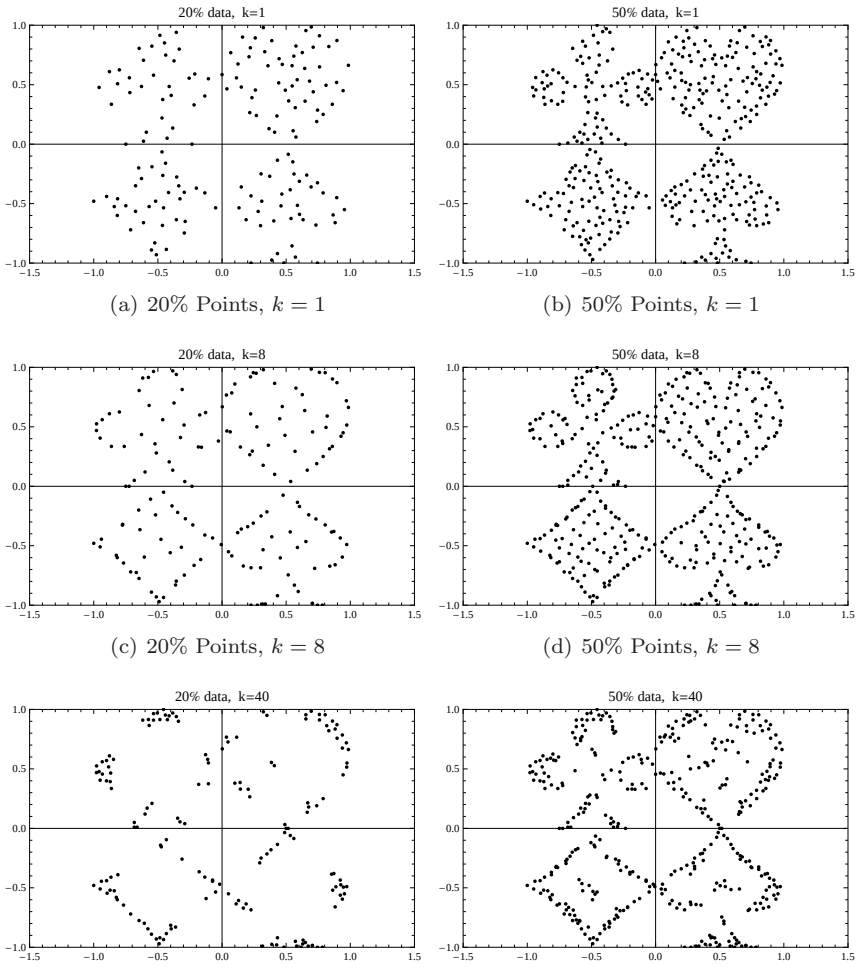
from the last point in the cluster. We compare the compression via the SMITS procedure and the compression based on the one-pass weight calculation on the **BAGEL** data with a non-uniform density, see Figure 2.12.

An efficient procedure of selecting 'space-filling' sub-sets of various sizes for problems of arbitrary dimensionality can have a significant impact on data preprocessing in modeling approaches, where the complexity of the resulting model depends on the number of points used in the modeling process. An example of such technique is kriging, that is recognized as a robust and effective interpolation method for relatively small and designed input-output data sets, but may have problems in convergence to a solution for very large data sets with non-uniform data density, see (Rennen, 2008) for examples of convergence problems.

2.6.3 Compression for response surface modeling

In cases, where data ranking or data compression are needed for the purposes of response surface modeling, the introduced iterative technique for sub-sampling should be used with weighting functionals adjusted to the input-output data - proximity, surrounding, remoteness, and non-linearity.

The rule-of-thumb that we are proposing is to use the non-linearity-based SMITS compression if the data is sampled reasonably uniformly in the input space, and the dimensionality of the input space is strictly smaller than the number of records. In other cases, the SMITS procedure should be used with surrounding or

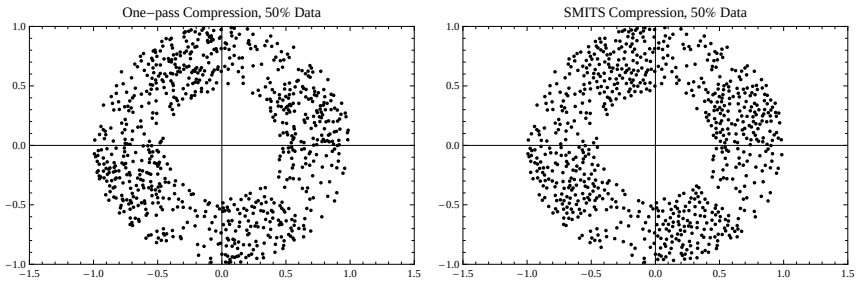


(a) 20% Points, $k = 1$ (b) 50% Points, $k = 1$

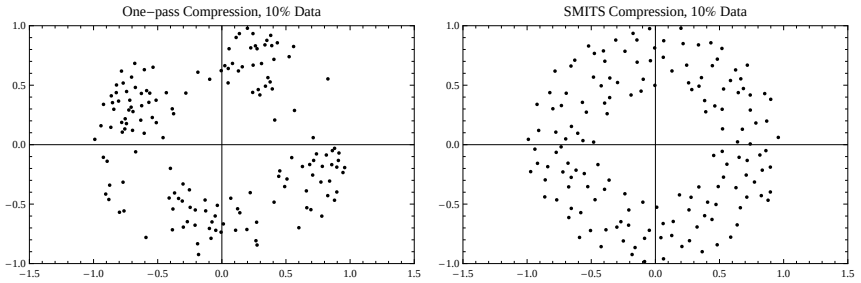
(c) 20% Points, $k = 8$ (d) 50% Points, $k = 8$

(e) 20% Points, $k = 40$ (f) 50% Points, $k = 40$

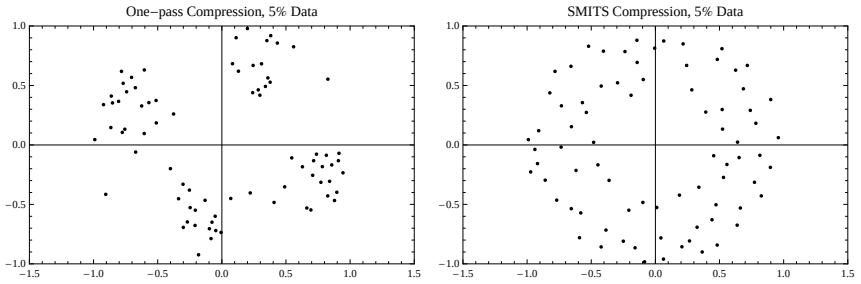
Figure 2.11: Space-filling compression of the SUITS data via the SMITS procedure with the remoteness-X weight. The plots represent the results of the compression of the **SUITS** data to 20% and 50% of the original size. We used three different neighborhood sizes to demonstrate the flexibility of the iterative elimination principle to the requirements of the user. The weight functional (remoteness-X) and the distance metric (Euclidean) were the same for all cases.



(a) One-pass Compression, 50% Points (b) SMITS Compression, 50% Points

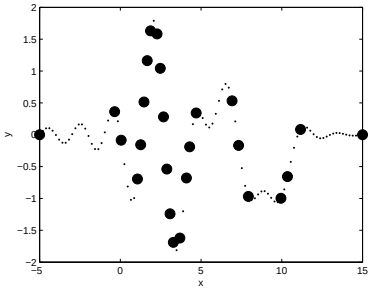


(c) One-pass Compression, 10% Points (d) SMITS Compression, 10% Points

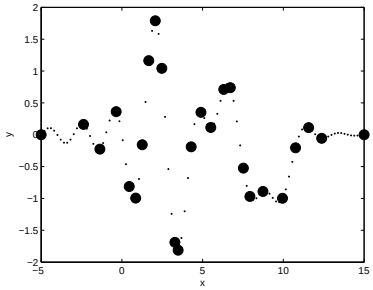


(e) One-pass Compression, 5% Points (f) SMITS Compression, 5% Points

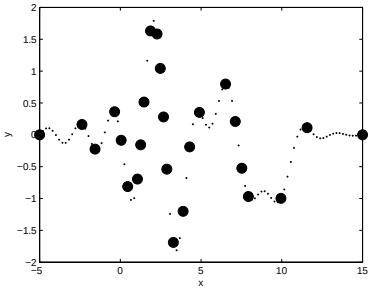
Figure 2.12: Compression of the BAGEL data, see Figure 2.10(b), via the one-pass remoteness-X weight calculation and via the SMITS procedure based on the remoteness-X weight.



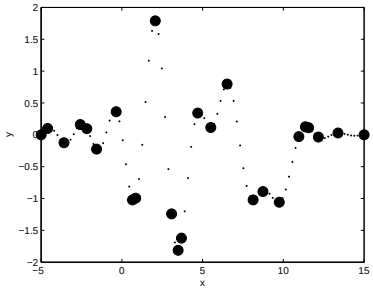
(a) Proximity, 25% Points, $k = 2$



(b) Surrounding, 25% Points, $k = 2$



(c) Remoteness, 25% Points, $k = 2$



(d) Non-linearity, 25% Points, $k = 2$

Figure 2.13: SMITS-based compression of the KSinc data with the proximity, surrounding, remoteness and the non-linearity weights and two nearest in the input space neighbors. The plots represent the results of the compression of the **KSinc** data to 25% of the original size.

remoteness weight.

The differences in compression of the **Ksinc** data set are shown in Figure 2.13. The SMITS procedure is applied to the **KSinc** set to rank it in the order of decreasing importance using proximity, surrounding, remoteness, and non-linearity weights. The first 25 points of each of the four ranked sets are plotted in Figure 2.13. We see that the subsets of the 25 'most important' records carry quite different information about the true response curve. The compression, based on the non-linearity weight, seems to be the most plausible, since the piece-wise linear interpolation through the produced 25 points describes the true response curve quite accurately (see Figure 2.13(d)).

2.7 Cumulative information content

This section presents a way for defining a measure for the cumulative information content of the input-output data set ranked in the order of decreasing importance with the SMITS procedure. Information content defined for a subset of the first m records of the ranked set can guide the user in selecting the compression threshold, and also can help to indicate the differences in compression, produced by four weighting functionals.

If during the SMITS procedure we archive a weight of each eliminated point, we can compute the cumulative sum of these weights for each elimination step in the order opposite to the order of elimination. If the weights are normalized by the total sum of the weights of the data set they will sum up to the number of records N. After normalization the resulting cumulative sum of eliminated weights can be interpreted as a cumulative information content of the data set ranked with the SMITS procedure.

In Figure 2.14 we plot the cumulative information content (CIC) of the Ksinc data ranked with four weighting functionals. A value on a curve at point m can be interpreted as a fraction of the information about the data contained in the first m samples of the ranked data set, $m = 1 : N$. The shape of the CIC curve can be considered as an indicator of compressibility of the Ksinc data ranked with the SMITS procedure.

If the decision is taken to compress the Ksinc data set to 25 points only, the CIC of the first 25 points of data ranked using the proximity, remoteness, surrounding and non-linearity weights will be 0.64, 0.70. 0.78, and 0.87 respectively (see Figure 2.14). This means, that among four subsets, the 25 'most important' points obtained using the non-linearity weight carry the highest fraction of information contained in the Ksinc data. Figure 2.13 with the actual results of the SMITS-based compression of the **KSinc** data to 25 points with proximity, surrounding, remoteness, and non-linearity weights carries the same message - the top 25 points of the compression based on the non-linearity weight is the only subset of four, that captures all the seventeen extrema of the **KSinc** function.

Figure 2.14 also illustrates, that if we are interested in selecting a smallest subset of essential points that contains at least 90% of the information of the **KSinc** data, we need to take the first 57 points of the set ranked using the

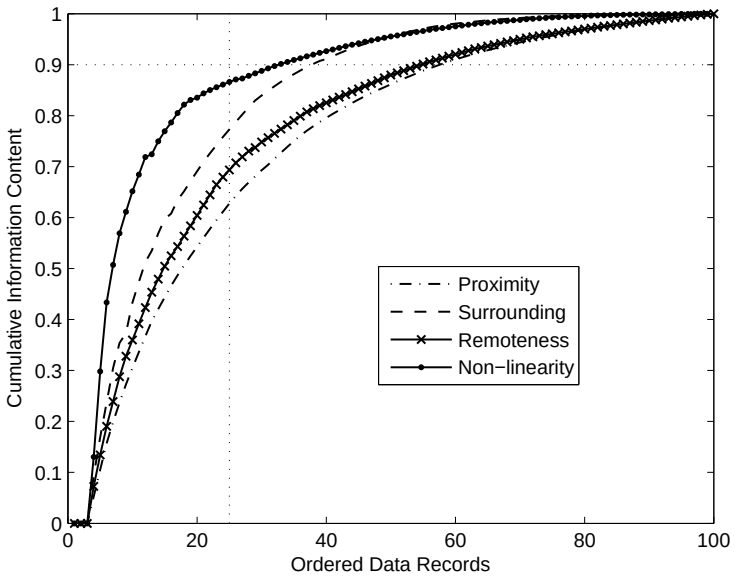


Figure 2.14: Cumulative Information Content as a characteristic of the compressibility of the data set ranked by the SMITS procedure. The plot represents the cumulative elimination weight of the the first m records of the **KSinc** data set ranked with the SMITS procedure for the proximity, surrounding, remoteness, and non-linearity weight on two nearest in the input space neighbors. We interpret it as the cumulative information content of the ranked data set. The steeper the shape of the cumulative information content curve, the more informative is the compression of the data ranked with a corresponding weight functional to a subset of the first m records.

proximity weight, the first 55 points of the data ranked using the remoteness weight, the first 39 points of the set ranked using surrounding weight, and only 33 points of the set ranked using the non-linearity weight. These observation suggests that the non-linearity weight might be a good choice of weight used for compression of a *relatively balanced* sets like **KSinc** data with the SMITS procedure.

2.8 Summary

The chapter has introduced four simple procedures for computing the importance of an input-output data record relative to its k nearest-in-the-input-space

neighbors and three procedures for computing weights of unlabeled input records. The importance, or information content, is defined based on the proximity, surrounding, remoteness or non-linear deviation from k nearest-in-the-input-space neighbors. The obtained weights can be either used for getting insight from the data before the modeling process, or for compression, or can be directly incorporated into the fitness function during modeling. Chapter 5 studies weighted symbolic regression in greater detail.

We observed that for very imbalanced and clustered data the four presented weighting functionals may not capture well the clusters of data that share the importance with respect to proximity or surrounding by k nearest-in-the-input space neighbors.

Reasoning that a part of a whole is important when its presence *or* its absence produces a large impact on the whole, we came up with a simple procedure of ranking the data by importance through the iterative pruning of points with the least obtained weight. The simple multi-dimensional iterative technique for sub-sampling (SMITS) takes the data set and the weighting functional as arguments and generates a ranking of the data set that can be used for constructing *balanced nested* subsets of an arbitrary size.

The SMITS procedure applied to one of the four weighting functionals allows sensible compression of large input-output data sets to smaller subsets of a similar information content. This feature may considerably enhance the analysis and modeling of large imbalanced data sets with highly non-linear response surfaces, if more modeling effort is applied to compressed subsets of data (see Chapters 3 and 5 for validation of this hypothesis).

An additional benefit of SMITS-based compression can be attained by applying it to other modeling methods besides evolutionary modeling. The rational point interpolations (Cuyt, 1993, 2003; Graves-Morris, 1981; Van Barel and Bultheel, 1990) and kriging are the meta-modeling techniques that are successfully used in practice for producing accurate global models of small- and medium-sized input-output data sets. The robustness and the complexity of solutions of both techniques may suffer dramatically if they are applied to very large input-output data sets (with more than two to three thousands of records). To increase the feasibility of convergence to a solution, kriging seeks for ways to compress the given (legacy) data to smaller subsets that are preferably space-filling in the input space (Rennen, 2008). The proposed SMITS-based compression applied

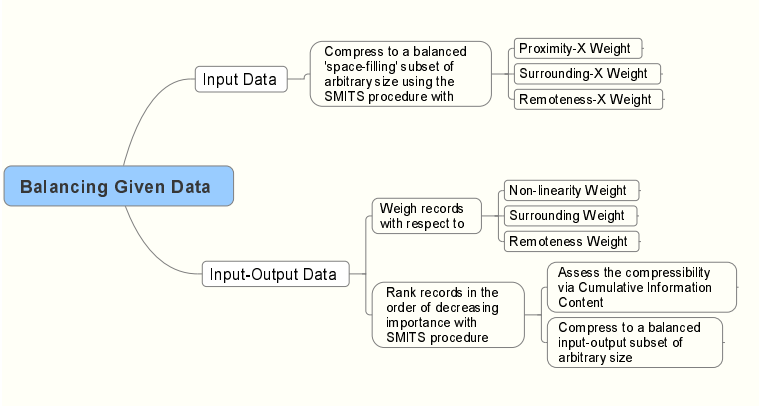


Figure 2.15: Summary of Chapter 1. Methods of balancing given data.

to the proximity or remoteness weights in the input space generates nested subsets that are sufficiently 'space-filling' for any subset size. We believe that for applications where no exact maximization of minimal distances is required (or possible), the SMITS compression becomes a competitive heuristic for creating space-filling subsets in the input space for arbitrarily large problems of arbitrary dimensionality[9].

For data sets of high dimensionality we strongly recommend using fractional distance metrics for finding nearest or nearest-in-the input space neighbors; see (Aggarwal et al., 2001; Doherty et al., 2004; François et al., 2007; Jin et al., 2003)).

Figure 5.10 summarizes the proposed methods to perform data weighting and compression.

If balancing is desired for response surface modeling, then the data should be weighted with either non-linearity, or surrounding, or remoteness weights. If for a set of d inputs it is possible to say that the average distances to $d+1$ nearest-in-the-input-space neighbors are similar (i.e. the data is not too imbalanced), then it is worthwhile to use the non-linearity weights. Otherwise, surrounding weights seem to be the second choice. After the data is ranked with the SMITS procedure, a 'plausible' rate of compression can be determined from the shape of

[9]Under condition that the impact of the curse-of-dimensionality is reduced by the use of an appropriate distance metric, which does not exhibit the concentration phenomenon

the cumulative information content.

We advise to compute the proximity, surrounding, and remoteness weights all together, and visually analyze their sorted profiles. If only a few points get excessively high proximity or surrounding weights, they are a subject for special attention. It is up to the user to decide whether to remove these points as outliers, or to inform the problem owner that the areas of the data space where these points are located require further sampling or analysis.

If the data set is too large and there is a specific interest in constructing a balanced subset that is 'space-filling' in the input space, then we recommend using the SMITS procedure with the remoteness-X, or proximity-X weights on one or $d+1$ nearest neighbors (d is the dimensionality of the input space). If the property of 'space-fillingness' is of interest, the input-output data should be weighted and ranked only in the space of inputs, as if data learning were unsupervised. We observed that compression based on the remoteness-X weights with one nearest neighbor gives satisfying results for various sets of various dimensionality.

We recommend using $k = d + 1$ nearest-in-the-input-space neighbors as a default setting for a general-purpose weighting routine. However, reducing the neighborhood size to one neighbor may be beneficial for dynamic environments, where the re-balancing should be done while spending minimal computational resources. In this case, we advise using surrounding or remoteness weight with one nearest-in-the-input-space neighbor, since the non-linearity weight is inapplicable.

Part II

Model Development

3

Model Evolution through Genetic Programming

This chapter presents basic principles of evolutionary search and their application to symbolic regression. Without claiming to be exhaustive, the Chapter focuses on symbolic regression via Pareto genetic programming and introduces definitions for various settings used for empirical experiments in this thesis.

3.1 Basic principles of evolutionary search

This Chapter starts Part II of this thesis devoted to Model Development. As presented in the Introduction, the results of the thesis are aligned with a framework of iterative model-based problem solving, where the task of problem solving is defined as identification of a symbolic regression model describing the behavior of the system in question. Problem solving is approached through iteration over the three stages: (1) data generation, analysis and adaptation, (2) model development, (3) problem analysis and reduction. In this thesis the stage of model development is in turn performed through iterative (evolutionary) search in the space of symbolic regression models constructed on given sets of basic functions, input variables, and constants.

Model development in the evolutionary framework consists of iterating through the three stages of: (1) model generation and modification, (2) model evaluation, and (3) model selection (see again Figure 1.5.1). Within each cycle of

problem solving, several cycles of model development are performed to identify regression models of sufficient quality, before they can be interpreted at the stage of problem analysis and reduction.

A set of alternative regression models considered at each step of the model development phase is called a population. Individuals of this population are undergoing many iterations of artificial evolution. The iteration steps are also referred to as generations.

The major expectation of performing the evolutionary search is that the balance between exploring new individuals and exploiting good (parts of) discovered individuals will improve in the quality of alternative solutions over generations. The magic of evolution (in this case of artificial evolution of formulae) lies in our observation that it does work very well across a wide range of problems.

The basic tenet of the evolutionary search (one that distinguishes it from random search) is persistent exploitation of discovered intermediate solutions, i.e. combination of good features of successful individuals for creation of (hopefully) even more successful progeny. One of the pitfalls of evolutionary search is an excessive exploitation of individuals, which results in 'inbreeding' — a premature zooming into a certain area of the search space, the loss of diversity among individuals (premature convergence, see (Langdon and Poli, 2002)), and stagnation of overall improvement leading to insufficient quality models.

To mitigate the risk of premature stagnation, the persistent exploitation is balanced by the continuous exploration of new alternative solutions. Without a doubt, the proper balance between exploitation and exploration is a cornerstone of successful evolutionary search. The mantra on the 'balance between exploration and exploitation' is attributed to John Holland (Holland, 1975), who developed genetic algorithms, which also need to exhibit this balance for optimization problems.

At the start of model development the initial population of regression models is either randomly initialized or suggested by the modeler (e.g. when the modeler wishes to re-use the information from the previous iterations of problem solving, or when a collection of pre-selected non-trivial regression models is available, as in (Korns, 2006)). The *Model generation phase* includes modifying a selected set of 'successful' regression models obtained at the previous step by re-combining or changing pieces of their structure. In the evolutionary

framework, model generation is often called *genetic modification* or *variation*, and consists of applying three main modification operators: re-combination of pairs of individuals (crossover), re-initialization of parts of a single individual (mutation), and copying an unchanged individual into the next generation (reproduction). Neither crossover nor mutation guarantee the improvement of quality (e.g. prediction accuracy) of individuals.

The *Model evaluation phase* includes estimation of quality of individuals with respect to a set of predefined criteria, like prediction accuracy and model complexity (more examples and definitions of evaluation criteria are given later in this chapter). Once the quality of individuals is determined, the *model selection phase* begins with applying a selection operator to the evaluated population.

The choice of a selection operator that picks out a subset of solutions that will be used for creating the new generation of individuals, is crucial in the evolutionary search, since it directly influences the balance of exploitation and exploration. Consequently it determines the speed of evolution, i.e. the number of iterations in which the desired level of solution quality can be reached (see (Banzhaf et al., 1998)). There are several types of selection strategies:

- Fitness proportionate selection,
- Rank-based selection,
- Tournament selection,
- Elite-based selection,
- Elite-based selection with elite preservation,
- Hybrid selection methods.

In fitness proportionate selection (introduced by Holland (1975) for genetic algorithms), the probability for an individual to be selected for propagation is proportional to the actual value of fitness of the individual. The pure fitness proportionate selection in GP without the pressure for diversity increases the risk of premature convergence.

The ranked-based selection operates on the orders of individuals ranked according to the fitness, rather than their actual values. The probability for individuals to be selected is a function of their rank in the population, and is scale-independent.

While the two mentioned selection schemes involve evaluation of a population and sorting of all individuals with respect to their fitness values, tournament selection is designed to avoid manipulations with the total number of population members and to reduce it to small tournaments. For each tournament, a small subset of individuals is randomly drawn from the population and evaluated, and a subset of the 'best' individuals (tournament winners) is selected according to a preferred meta-selection scheme. Despite the fact that tournament selection has to be applied multiple times to collect a sufficient amount of winners for creating the next generation, these computations can easily be parallelized and therefore are preferred over, for example, ranked-based selection.

In elite-based selection, a special subset of *elite* individuals is determined and propagation rights are distributed uniformly over the elite members.

Archive-based selection is closely related to elite-based selection, but involves a preservation of the elite set of individuals over the entire search process. In this case the selection operator is applied to both current population members and old archive members. The selected individuals are modified to create the next population. If improvement is observed in the new population compared with the archive, the archive is updated with out-performing individuals.

The archive represents a memory of the search system that contains the 'best-so-far' solutions (actual individuals, not only their cashed fitness values) at all iterations steps. Maintenance of an archive is a powerful way to pursue exploitation in a controlled fashion. The impact of archiving on the performance of the evolutionary search can be compared with the impact of finite memory on the computational power of an automaton (for latter see (Kudryavtsev, 1995)).

The crucial significance of archiving for stochastic iterative search has been widely recognized in the area of multi-objective evolutionary algorithms after development of effective elite preservation algorithms for multi-objective optimization, like SPEA (Zitzler and Thiele, 1999) and NSGA-II (Deb et al., 2002).

The Pareto genetic programming system studied in this thesis also uses archiving for enhanced exploitation with the hybrid selection scheme described in the next section in more detail.

After a selection operator is applied to population members (or population and archive members in case of archive-based search), the selected individuals proceed to the model generation stage, where modification operators are applied

to them to create the population for the next iteration step.

There are several ways to built the new population that differ in the number of offspring and the style of their insertion into the new population. The differences are related to the desired intensity of information re-use. E.g., the old population can be completely discarded and substituted by the new population (in this case the number of off-spring equals the population size). Due to the fact, that the improvement through variation is not guaranteed, discarding the old population is a risky endeavor if some form of archiving is not performed. If the pool of offspring is larger than the population size, then some selection pressure should be applied to condense the number of alternative solutions to a new population.

This section has presented a very simple overview of three main stages of the evolutionary search, that have to be repeated over a required number of generations or until a perfect solution is found. It is necessary to mention that evolutionary search is a stochastic iterative search method performed in a huge (in general infinite) search space, and therefore it requires multiple replications to be performed to increase the confidence in the results. Nothing has been said so far about the quality measures used for determining successful individuals at the model selection stage, and for monitoring the performance of the evolution. Since the quality measures are directly related to the properties of individuals and also to the representation of individuals, we discuss them together in the next section, where we focus on Pareto genetic programming.

3.2 Basic scenario of a genetic programming run

3.2.1 Model representation and phenotype

There is a multitude of ways to represent symbolic expressions of a given set of input variables. The task of symbolic regression is to identify expressions in the following form:

$$\hat{f} = \mathcal{M}(\mathcal{X}, \mathcal{F}, \mathcal{C}), \tag{3.1}$$

where $\mathcal{M}$ is a symbolic model representing a valid mathematical expression on the alphabet of input variables $\mathcal{X} = \{`x_1', `x_2', \ldots, `x_d'\}$, basic function operators $\mathcal{F}$, and constants $\mathcal{C}$.

A set of basic operations contains functions that have one or two arguments.

Typical representatives of the set are the standard arithmetical functions: addition, subtraction, multiplication and division. They may also include power, trigonometric, logarithmic, exponential, and logical functions. The set of constants can either be fixed to a certain set of random numbers from a certain interval, or it can be infinite, and contain arbitrary random constants drawn from a certain interval, as in Pareto GP.

In this thesis only real-valued data is used, and by valid mathematical expression we understand a related mapping $\hat{f}\colon \tilde{X} \subset \mathbb{R}^d \to \mathbb{R}$, where $\tilde{X}$ is a feasible region of input space. Usually $\tilde{X}$ is supplied by the data owner, or is determined from the maxima and minima of the input variables.

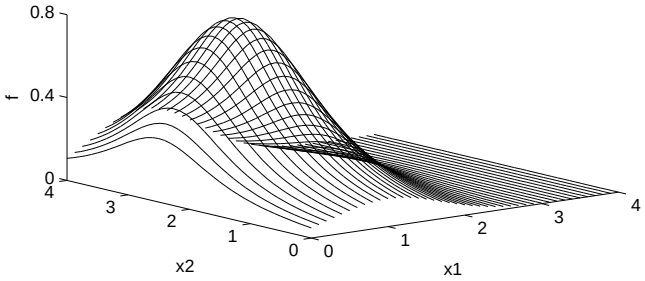
Any formula $\hat{f} = \hat{f}(\mathbf{x})$, where $\mathbf{x} = (x_1, x_2, \ldots, x_d)$, determines a response surface in $\mathbb{R}^{d+1}$. Symbolic regression studied in this thesis assumes that the mapping of formulae $\hat{f} = \hat{f}(\mathbf{x})$ to response surfaces is surjective, i.e. every response surface can be explicitly expressed as a closed-form formula on the input variables, which may not be true. In fact, searching for implicit relationships among input and output variables, e.g. in the form of $\hat{\hat{f}}(\mathbf{x}, y) = 0$, with a posteriori application of numeric methods to solve produced equations for the response, may be beneficial for modeling hard problems, where the explicit relationships may not exist.

Anyway, the mapping of formulae to response surfaces is certainly not injective, since different formulae may produce identical response surfaces. In principle, for each response surface there is an infinite number of expressions defining it. This leads us to the definition of two important abstractions in the evolutionary search - a genotype and a phenotype of an individual.

A phenotype is the observable entity of the individual, its functional morphology. In symbolic regression, this entity is the actual relationship, i.e. the response surface. A genotype, or more correctly genome[1] is the information medium, or a code, characterizing the individual. In symbolic regression, this code is the actual symbolic expression (3.1).

In the evolutionary framework for symbolic regression, genomes can be represented in various ways: as linear strings, tree structures, directed graphs, etc. In symbolic regression via genetic programming individuals are usually represented by structures of different size (this is a general difference between genetic programming and genetic algorithms, where the first was designed to

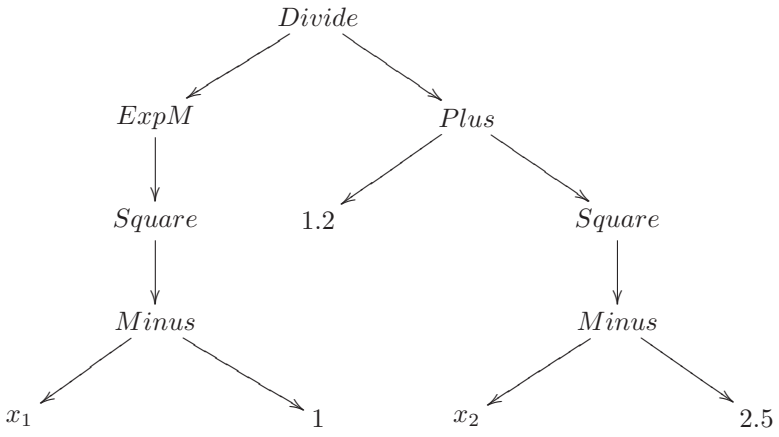
[1] A genotype characterizes a type of species, while a genome characterizes a single individual.



(a) Response Surface - Phenotype

$$f = \frac{e^{-(x_1-1)^2}}{1.2+(x_2-2.5)^2}$$

(b) True Formula



(c) Tree-based Genome of the Perfect Solution

Figure 3.1: Genotype and phenotype in symbolic regression

operate on trees of various size and shape (Koza, 1992), and the latter was designed to operate on binary strings of fixed length (Holland, 1975)).

All experiments of this thesis are performed with a tree-based representation of individuals. Or more exactly, an individual in this thesis is defined as a binary directed tree, with terminal nodes labeled by symbols from the alphabets of variables or constants, and inner nodes labeled by symbols from the alphabet of basic functions. The binarity of the tree implies that only unary or binary basic functions are used (see Figure 3.1 for an example).

In general, the allowed arity of basic functions (the number of variables that the functions take) exerts a large influence on the expressiveness of the representation, and compactness of individuals. High arities potentially allow very compact expressions for individuals, e.g. if summation of arity four is allowed, expression $Plus[Plus[Plus[a,b],c],d]$ can be rewritten as $Plus[a,b,c,d]$. On the other hand, large arity has to be controlled when the trees are randomly initialized.

There are two most common methods for initializing trees - full and grow (see Koza (1992)). In Pareto GP the trees are initialized from the root towards the leafs with the 'grow' method (see (Koza, 1992)), up until the maximum allowed size (e.g. the depth) is reached. In the grow method the labels for the nodes are assigned randomly from the set of functions and terminals (variables and constants), according to some pre-defined probabilities. The used version of Pareto GP contains the probability to assign a function label to the next node, and the probability to assign a variable label to the next terminal node. The grow method generates tree structures of diverse size and shape[2]. The shape of the trees does not only depend on the probabilities to assign function labels as the next node, but also depends on the ratio between the number of unary and binary functions in the basic function set.

Despite the fact that function labels are assigned uniformly at random from the set of basic functions, the modeler can bias the function distribution, and hence the shape of the trees, by modifying the function alphabet $\mathcal{F}$. The default functions in the basic set of Pareto GP palette are $\{Plus, Minus, Times, Divide, Negate, Exp, ExpM, Square, Power_C, Shift_C, Scale_C\}$, where $Power_C$, $Shift_C$, $Scale_C$ are unary functions with a dimensionless constant parameter C,

[2]In the full method, only function labels are assigned to the nodes of the tree up until the maximum size is attained, and then terminals are assigned randomly from the terminal set.

corresponding to rasing the argument to the power C, shifting the argument by C, and scaling the argument by C respectively. Note that the first four functions are binary, and the other seven are unary.

Such introduction of unary functions with parameters is a powerful way to enhance exploitation of constants in GP, and to reduce the risk of disrupting important building blocks.

If is interesting to note, that at the beginning of the research project that led to this thesis, analysis of the early disappearance of constants from the population and the archive in Pareto GP was one of the research goals. However, after a mere introduction of the $Shift$ and $Scale$ operators into the basic function set, the problem of lacking constants in final solutions miraculously disappeared.

It can be speculated that by introducing unary operators with constant parameters like $Power_C$, $Shift_C$, $Scale_C$ we:

1. tie constants to associated sub-expressions;

2. avoid disruption of constants and sub-expressions by crossover;

3. potentially seed individuals with more constants without running the risk of increasing the non-linearity of individuals by accidental substitution of constant nodes by subtrees in case of crossover.

Also note, that besides having a $Power_C$ function ($Power_C(x) = x^C$), the basic function set always contains a $Square$ function ($Square(x) = x^2$) to drive the system towards simpler exponentiation.

All functions are unprotected as in (Keijzer, 2003) and (Vladislavleva, 2005). In all experiments constants of the set $\mathcal{C}$ are randomly drawn from a real interval $[-10, 10]$ as many times as necessary.

Binary tree representation is the most common 'academic' fashion to define individuals in genetic programming, partly because it is the oldest one, and theoretically most tractable. However, the quest for the best representation is still in progress, since representation directly influences the speed of evaluation and generation of individuals.

In general, three requirements for a good representation can be formulated:

1. A good representation should allow very fast evaluation of an individual. E.g., traversing a tree may be inferior in speed to evaluating a pre-compiled

linear genome as in machine code representation, for example as in (Banzhaf et al., 1998; Nordin, 1994).

2. A good representation should allow relaxed modification of individuals. E.g., in tree representation one point mutation has to be performed very accurately, so that e.g. a node label corresponding to a unary operator is not accidentally mutated by a label corresponding to a binary operator. This checking for legal modification introduces unwanted overhead. An example of an alternative solution is the Karva-representation by Ferreira (2001). The latter uses linear genomes of fixed size that are transcribed into trees of different size and shape. An interesting property of this representation is the existence of a 'tail' of terminals, which allows to randomly initialize and modify labels in the strings, without the risk of producing trees coding invalid mathematical expressions.

3. A good representation should also be such that the modification operators are not too disruptive for low-order transformations conferring high potential to the host individual. In other words, a good representation should preserve building blocks[3]. E.g., with respect to disruptiveness by one point crossover, a standard tree representation preserves the entire branche rooted in crossover points, while in Karva representation the branches of the transcribed trees rooted in crossover points, are in principle re-modeled.

3.2.2 Model generation and genetic operators

As mentioned in the previous section, in Pareto genetic programming as in other GP variations, an initial population can be either randomly re-initialized, or seeded with the results of previous runs. In all experiments of this thesis populations are initialized at random.

To reduce the risk of premature convergence through inbreeding, Pareto GP system applies a cataclysmic re-initialization to all population individuals once in a while. The number of generations, over which the population can evolve before such re-initialization happens is called a *cascade*. In this thesis a cascade

[3]Everywhere in this thesis I mean by building blocks low order transformations, causing a high fitness jump to the hosting individual. Loosely speaking, building blocks of interest are such transformations that if used as meta-variables may lead to a linear, or a low-order relationship of the response variable to the input meta-variables.

is ten generations long (this means that the population is re-initialized randomly at generations $1, 10, 20, \ldots$, etc.). Short cascades of only 10 generations seem to improve modeling performance compared with longer cascades, probably because exploitation is guaranteed due to maintenance of the archive, and the use of population individuals has an explorative nature.

The existence of the archive in Pareto GP allows to perform only two genetic operators for model modification - crossover and mutation (no copying). The size of the offspring pool is equal to the population size. The number of individuals, generated by crossover is controlled by the *crossover rate* parameter. In all experiments of this thesis 95% of individuals in the new population are generated by crossovers between members of the archive and the old population (see next subsection for details on the selection scheme). For recombination 'balanced' one-point crossovers are used, where in one individual the crossover point is selected at random, and in the second individual the crossover point is selected from the layer similar to the layer corresponding to the crossover point of the first parent. Then the subtrees with roots in selected nodes are swapped among parent individuals, and two new individuals are created.

Such 'balanced' crossover for symbolic regression problems reduces the speed of complexity growth, observed when the standard uniform crossover is used, and also reduces the crossover bias described by Keijzer and Foster (2007) and Poli et al. (2007). Despite the fact that balanced crossover is a 'quick fix' to the problems associated with the standard uniform crossover on binary trees, further research is needed to analyze its appropriateness for the class of symbolic regression problems. Xie et al. (2007) study the strategies for controlling the depth of crossover points on various GP systems in three problem domains, and report that controlling the depth of crossover does not guarantee a significant improvement in general, and its performance also depends on the stage of evolution.

The crossover rate also determines the fraction of individuals generated by point mutations. In this thesis 5% of progeny are generated by point mutation applied to archive members. As described in the previous section, mutations are protected; i.e., a terminal node can be mutated into a terminal node, a functional node - only into a function of the same arity. One more internal parameter controls the rate of mutation on terminals. In this thesis it is equal to 3% of all mutations. In principle, for problems with very large number of input

variables (hundreds or thousands), the user may want to increase the mutation rate on terminals to increase the confidence of the variable selection procedure; see Smits et al. (2005), and Chapter 8.

3.2.3 Model evaluation and quality measures

Fitness as the quality of the phenotype

One of the main and mostly used quality measures of a symbolic expression is the quality of its phenotype, i.e. prediction error, defined as a measure of deviation of the response surface from the observed data records. In Chapter 2 of this thesis two common error measures are introduced - the mean squared error (mean sum of squared errors) and the correlation coefficient of predicted and observed response.

$$MSE(\mathbf{y}, \hat{f}(\mathbf{x})) = \frac{1}{N}\sum_{i=1}^{N}(y^i - \hat{f}(\mathbf{x}^i))^2, \tag{3.2}$$

$$R(\mathbf{y}, \hat{f}(\mathbf{x})) = \frac{cov(\mathbf{y} \cdot \hat{f}(\mathbf{x}))}{std(\mathbf{y}) \cdot std(\hat{f}(\mathbf{x}))}, \tag{3.3}$$

where $(\mathbf{x}^i, y^i)$, $i = 1 : N$ is a training set of N input-output records. To avoid confusion, subscripts of input variables denote the variable number, i.e. the column number in the data matrix, superscripts denote the record number, i.e. the row in the data matrix. The size of the data matrix is $N \times (d + 1)$, where d is the dimensionality of the input space.

Whatever the error measure is, it is important to ease the identification task to a GP system and let the system focus the search effort on discovering the right structure rather than the right scaling. It seems that the mean squared error is a more informative measure for evaluating *bad* solutions (e.g. at the beginning of the evolutionary run), than a correlation coefficient is. The last measure is designed to explain the linear dependency of the predicted and observed response, and can therefore be meaningless when the actual errors are large. In this case the mean squared error can still produce a meaningful ranking of individual predictions. Since the mean squared error is scale dependent (even if normalized by the standard deviation of the response; see e.g. (Kleijnen and Sargent, 2000)), it is crucial to scale the predicted and observed responses to the same ranges or

to the same deviations, before computing the error.

The thesis (Keijzer, 2002) refers to such scaling as to prediction wrapping, where instead of a generated predicted output $\hat{f}$ a scaled (wrapped) output $\tilde{f} = a\hat{f} + b$ is considered. The optimal choice for coefficients a and b is the choice of the scale and the intercept of the least-square regression line constructed in the space of predicted versus observed response. Optimally chosen coefficients a and b imply a reduced (the minimal) mean squared error of the scaled prediction $\tilde{f}$ (by definition of the least-square regression line). The formulae for a and b are given below:

$$a(\hat{f}, \mathbf{y}) = \frac{\sum_{i=1}^{N} \left(y^i - \overline{y}\right) \left(\hat{f}(\mathbf{x}^i) - \overline{\hat{f}(\mathbf{x})}\right)}{\sum_{i=1}^{N} \left(\hat{f}(\mathbf{x}^i) - \overline{\hat{f}(\mathbf{x})}\right)^2}, \tag{3.4}$$

$$b(\hat{f}, \mathbf{y}) = \overline{y} - a(\hat{f}, \mathbf{y}) \cdot \overline{\hat{f}(\mathbf{x})}. \tag{3.5}$$

The paper (Keijzer, 2003) suggests to use the scaled predicted response instead of a computed predicted response for calculating fitness of GP individuals, where the scaling coefficients are obtained by formulae (3.4) and (3.5). When this is done, and a *scaled predicted response* $a(\hat{f}, \mathbf{y}) \cdot \hat{f}(\mathbf{x}) + b(\hat{f}, \mathbf{y})$ is used instead of $\hat{f}(\mathbf{x})$ for training error calculation, e.g. according to the formulae (3.2) and (3.3), we will say that error evaluation is performed with the *optimal scaling*.

Computation of coefficients a and b according to the formulae (3.4) and (3.5) can be done in $O(N)$ arithmetic operations per individual, however the actual operations on vector $\hat{f}(\mathbf{x})$ are quite intensive.

It is not necessary to store the optimal scaling coefficients for all individuals during the evolution. The main goal of computing them is to scale the predicted output to the same mean and standard deviation as the observed output, to focus the effort of the GP system on identifying the structure of the relationship, rather than on identifying the scaling first and then the structure. One way to avoid intensive calculations of the deviations of the predicted response for a GP individual in formula (3.4) and still pursue the same goal, is to consider an alternative, sub-optimal scaling of the predicted response.

All experiments of this thesis, using the mean squared error as a fitness function, employ the alternative scaling scheme within the evolution, but compute the optimal scaling coefficients for final solutions at the end of the

evolutionary run according to formulae (3.4) and (3.5).

The alternative scaling scheme is the same as in (Smits, 2001) and involves scaling all predicted responses together with the observed response to the interval $[0, 1]$, and then computing the error on scaled vectors.

As mentioned in the introduction, and as described in the next subsection in detail, Pareto GP performs evolutionary search under simultaneous minimization of two model quality characteristics - prediction error and model complexity.

For the sake of convenience and generality a bi-objective optimization of model fitness and model complexity is formulated as a minimization problem. For this, the error functions used for computing the prediction error, are normalized.

When mean squared error is used as a basis for error calculation, the fitness of individual $\hat{f}$ on training data $(\mathbf{x}, \mathbf{y})$ in this thesis is defined as:

$$E_{MSE}(\mathbf{y}, \hat{f}(\mathbf{x})) = 1 - NMSE(\mathbf{y}, \hat{f}(\mathbf{x})), \tag{3.6}$$

$$NMSE(\mathbf{y}, \hat{f}(\mathbf{x})) = \frac{1 - MSE(s(y), s(\hat{f}(\mathbf{x})))}{1 + MSE(s(y), s(\hat{f}(\mathbf{x})))}, \tag{3.7}$$

$$s(\mathbf{y}) = \frac{\mathbf{y} - \min \mathbf{y}}{\max \mathbf{y} - \min \mathbf{y}}. \tag{3.8}$$

These equations imply that the normalized mean squared error $NMSE$ is bounded by $[0, 1]$ with the perfect solution corresponding to $NMSE = 1$, and hence with $E_{MSE} = 0$.

When the correlation coefficient is used as a basis for error evaluation, the corresponsing fitness measure for GP individual $\hat{f}$ on the training set $(\mathbf{x}, \mathbf{y})$ is defined as:

$$E_R(\mathbf{y}, \hat{f}(\mathbf{x})) = 1 - R^2(\mathbf{y}, \hat{f}(\mathbf{x})). \tag{3.9}$$

Individuals with infinite or complex-valued training errors are removed from the populations and do not take part in the evolution. This implies that the archive individuals have finite real-valued prediction errors at all times.

Since symbolic regression is an identification method, prediction accuracy of the developed models is an important but not the only measure to optimize. Especially since the class of potential solutions describing the (true) response surface is broad, choosing the solutions of optimal complexity is an important

challenge (see, for example, (Cherkassky, 2002; Cherkassky and Mulier, 1998; Smits and Kotanchek, 2004)). Below a way to measure the complexity of tree-based individuals is presented.

Complexity as a quality of the representation

In classical GP applications (Banzhaf et al., 1998; Koza, 1992; Langdon and Poli, 2002; Poli et al., 2008) survival of the fittest is the most common single criterion to find the optimal solution. As mentioned in the Introduction, model interpretability and generalization capability are as important for model deployment as is its prediction accuracy. In real-life applications data is often corrupted by noise, and a good generalization of a model to 'smoothen' the noisy response is vital. Besides, for noisy problems with unknown true response THE optimal solution does not exist.

Often models with high goodness of fit look so obscure that it becomes infeasible to convince process engineers to implement them for controlling real on-line processes. In all these cases, simpler credible models with a lower level of fitness are always preferred over complex ones. Moreover, limiting the complexity of models may be crucial in avoiding over-fitting of data and also modeling the process noise. Too complex models are difficult to use, whereas too simple models may give poor prediction. For classical modeling techniques, model complexity is controlled by *a priori* knowledge of the process and the true underlying relationship; see (Cherkassky and Mulier, 1998). In these cases, parsimony pressure is introduced in the fitness function, and a resulting composite fitness function is defined as a linear combination of prediction error and a complexity term (the latter is called a regularization coefficient).

Irrespectively on how accuracy and complexity are combined in the optimization process, as a composite fitness function, or as independent objectives (yet, preferably as independent objectives), the complexity has to be defined, before it can be optimized.

Until now, almost all complexity measures considered by the GP community have addressed the structural complexity of an individual, i.e the complexity of the genome.

There is a variety of intuitive measures for determining the size of structures operated by a tree-based GP:
1) The number of nodes in a tree;

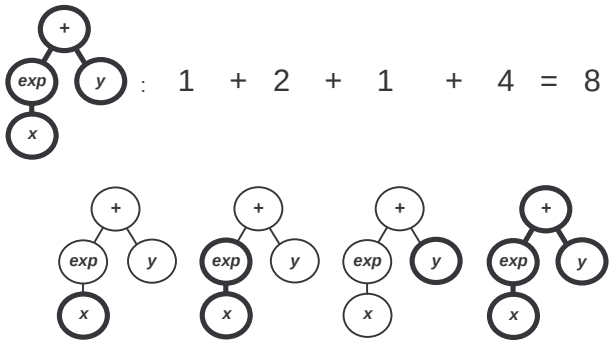


Figure 3.2: Expressional complexity of a tree model is the total number of nodes in all sub-tree models.

2) The number of layers in a tree;
3) Path length, etc [4].

To measure the size of individuals, this thesis uses a so-called expressional complexity (see Smits and Kotanchek (2004)). This measure is determined by the sum of the number of nodes in all subtrees of a given tree. It favors the flatter trees (i.e., trees with fewer layers and, hence, with fewer nested functions) over deep unbalanced trees (in the case of an equal number of nodes). An example of such a case is shown in Figure 3.2. The expressional complexity can be interpreted as the size of the model obtained by substituting all inner functions of the model by their function bodies. Keijzer and Foster (2007) refer to this complexity measure as to a visitation length, show that it is a close relative of the path length, and provide a thorough review of its mathematical properties.

The expressional complexity measure has been used in at least two commercial GP systems - in the ParetoGP Toolbox for Matlab (Dow proprietary product) and in the DataModeler add-in for Mathematica. Optimizing the structural complexity of evolving models together with the goodness of fit has been shown to produce compact solutions that are interpretable and reliably accurate within the training range (see, e.g., (Kotanchek et al., 2006; Smits and Kotanchek,

[4]H. Iba in Iba et al. (1994) uses a minimal description length principle, which is not explicitly a complexity measure, but is related to a representation, and can be considered as a complexity measure.

2004)). The pressure for compactness during evolutions indeed contributes to generalization properties of solutions, which agrees with (Rosca, 1996; Zhang and Mühlenbein, 1995). However, after extrapolation, these solutions may still demonstrate unwanted behavior, caused by over-fitting. The cause is that shorter solutions are not necessarily the smoother solutions, as noted by (Cavaretta and Chellapilla, 1999; Domingos, 1999; Gagné et al., 2006; Vladislavleva et al., 2008). Chapter 4 addresses this issue and introduces an alternative genome-phenotypic complexity measure, called the order of non-linearity, together with related selection strategies.

3.2.4 Model selection and complexity control

Penalizing models for high complexity is a natural way to control bloat, i.e. excessive growth in the size of GP individuals without improvements in fitness; see (Banzhaf and Langdon, 2002; Blickle and Thiele, 1994; Gustafson et al., 2004; Koza, 1992; Langdon and Poli, 1998; Langdon et al., 1999; McPhee and Miller, 1995; Nordin and Banzhaf, 1995; Poli et al., 2007; Soule et al., 1996; Streeter, 2003).

Since bloat is a widely recognized problem for successful evolution in GP, researchers have extensively studied evolutions under parsimony pressure and their relationship to bloat (see Gagné et al. (2006); Soule and Foster (1998); Soule and Heckendorn (2002); Zhang and Mühlenbein (1995)). Parsimony pressure, defined as a linear term added to the fitness function, causes GP to perform well on some problems, Blickle (1996); Soule et al. (1996); Zhang and Mühlenbein (1995), and less well on others, Koza (1992); Nordin and Banzhaf (1995). Soule and Foster (1998) showed that the linear coefficients in a composite fitness function relating numerical fitness and the structural complexity, can be used as a good indicator of the performance of a GP population. However, the search for a good combination of these coefficients requires some intuition and empirical testing.

We are certain that when no *a priori* information about the problem is known, the measure for parsimony pressure has to be optimized *individually* and *simultaneously* with numerical fitness. This fosters an intelligent trade-off between model simplicity and model accuracy. Optimizing complexity and accuracy in a truly multi-objective way will exempt us from making risky

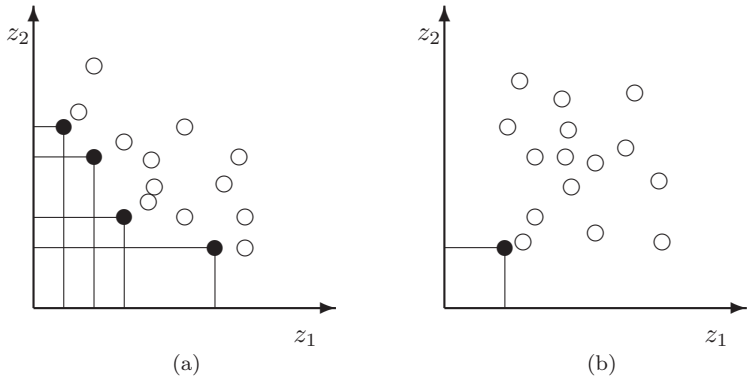


Figure 3.3: Pareto fronts in a bi-objective minimization problem.

assumptions about the exact relationship between complexity and accuracy, and will therefore not bias the search.

With such bi-objective selection, the GP system is pushed to produce both accurate and simple individuals, from which the best ones form a Pareto frontier (or Pareto front) – a set of optimal trade-offs in the two-dimensional performance space of complexity and accuracy (see Figure 3.3).

The used implementation of symbolic regression via Pareto GP adopts an archive-based selection strategy to select good models and endow them with improved propagation rights. The definition of what is good is based upon the concept of dominance in a space of selected optimization objectives (see the definition below). Propagation rights are granted to all archive members irrespective of their relative numerical goodness of fit. Archive members generate offspring for the next evolutionary step, and these new models are then used to optimize the archive. When the iteration process is terminated, the set of final GP solutions is determined by the archive at the last iteration step. An excellent description of the elite-based strategy with archiving for multi-objective evolutionary computation is given in (Fonseca and Fleming, 1995; Laumanns et al., 2002). Figure 3.4 reproduces this scheme with minor modifications, reflecting the selection in Pareto GP.

The process of updating the archive at each generation employs the concept of

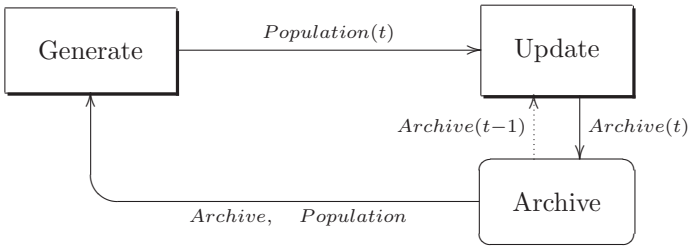


Figure 3.4: Generic scheme for archive-based evolution. Block GENERATE **produces a new population of individuals** $Population(t)$ **at generation** t **(**$t > 1$**) by applying modification operators to individuals of population** $Population(t-1)$ **and archive** $Archive(t-1)$ **from the previous generation. The new population** $Population(t)$ **is then used by block** UPDATE **to create a new archive** $Archive(t)$ **containing the elitist individuals from the union of the** $Population(t)$ **and** $Archive(t-1)$**.**

Pareto-optimality in a set of q selected optimization objectives $\Theta = z_1, z_2, \ldots, z_q$ (see Fonseca and Fleming (1995); Zitzler et al. (2003)).

Consider a minimization problem, where smaller values of all objectives $z_1, \ldots, z_q$ are preferred. Let Z be an operator, mapping a space of GP individuals F into the space of optimization objectives $\Theta \subset \mathbb{R}^q$, such that

$$Z : \hat{f} \in F \mapsto (z_1(\hat{f}), \ldots, z_q(\hat{f})) \in \Theta.$$

Solution $\hat{f}_1$ is said to (strongly) dominate solution $\hat{f}_2$ in the objective space Θ if the following condition holds:

$$\{\forall j \quad z_j(\hat{f}_1) \leq z_j(\hat{f}_1), \quad and \quad \exists j_0 : z_{j_0}(\hat{f}_1) < z_{j_0}(\hat{f}_1)\}, \quad j, j_0 = 1 : q.$$

The Pareto front is the set of objective vectors of Θ, which are non-dominated by any other objective vectors. In other words, the Pareto front is a set of all optimal trade-offs in the objective space Θ, which does not depend on the scaling of objectives. Figure 3.3 gives two examples of the Pareto fronts for a bi-objective minimization problem.

In Pareto GP, presented in (Smits and Kotanchek, 2004), the two minimization objectives are prediction error and expressional complexity. Thus every individual can be mapped into complexity versus error objective space, such that $Z : \hat{f} \mapsto \Big(E(\mathbf{y}, \hat{f}(\mathbf{x})), \Phi(\hat{f})\Big)$, where $E(\mathbf{y}, \hat{f}(\mathbf{x}))$ is defined by formula (3.6) or (3.9), and

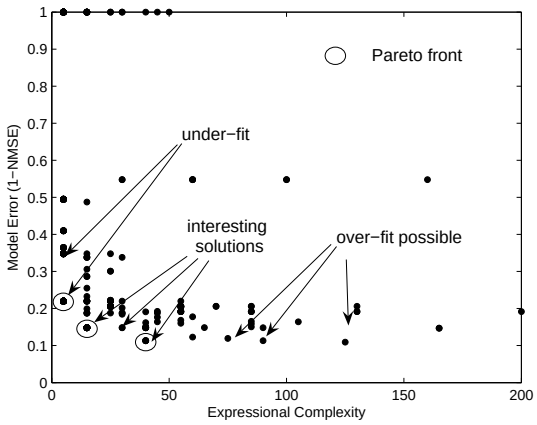


Figure 3.5: Pareto front and over-fitting. For the shown set of GP individuals, plotted in the objective space (model expressional complexity vs. model error $(1 - NMSE)$, the non-dominated set consists of only three individuals. Points with low model error and very high expressional complexity are the first suspects of over-specialization. The best-of-run models are almost always at the bottom-right corner of the plot (i.e. are almost always over-fitting). Points with too low expressional complexity may be too compact to describe the data, and, hence, have high errors. Note that we want to focus the search on the area around zero (low error, low complexity).

$\Phi(\hat{f})$ denotes expressional complexity (the sum of nodes in all sub-trees of the tree, defining $\hat{f}$). Another example of a Pareto front corresponding to a small population of GP individuals is given in Figure 3.5.

Plotting individuals in the objective space is a useful way to visually assess their overall quality, and identify a subset of the the most interesting individuals located in the region around zero. The task of multi-objective optimization problems is to push the Pareto front towards zero, where both optimization objectives are optimized. Population members which correspond to the Pareto front points in the objective space, are in principle the best candidate solutions for model selection, since they exhibit the optimal and incomparable combinations of complexity and accuracy. However, since the number of points on the Pareto front can be small (see (Vladislavleva, 2005)), and it varies significantly over the iterations, model selection focused on the strong Pareto dominance can suffer from a loss of diversity.

The non-dominated sorting algorithm by Deb et al. (2002) is designed to sort individuals in the order of increasing non-dominance. For this goal, first all

individuals on the Pareto front are identified, assigned a rank of 1, and removed from the main set; then individuals on the Pareto front of the remaining subset are identified, assigned a rank of 2 and removed from the main set. The procedure is repeated until a desired number of low-ranked individuals is extracted from the population. This algorithm can be implemented in $O(n^2)$ operations, where n is the number of individuals in the main set (Deb et al., 2002).

This thesis uses the non-dominated sorting algorithm in the complexity versus error space to create an archive of a constant size. Figure 3.6 presents an example of initializing an archive by the 50 best individuals from a population of 100 individuals at the first generation of a Pareto GP run.

After the archive is initialized, it is used to generate the new population according to the scheme in Figure 3.4. The population for the next generation is created by crossovers between the current population and the archive, and by mutations of the current archive. For the case of crossover an additional tournament selection operator is applied to archive individuals and to population individuals to select parents for the crossover. Experiments presented in this thesis use standard fitness-based single winner tournaments of size five for population and size three for the archive. Two-objective tournaments, however, may be more appropriate for selecting parents for crossover, since by design they produce multiple winners with guaranteed diversity, and in case of Pareto tournaments in the complexity-accuracy objective space, focus selection on the most interesting subspace of solutions that are both simple and accurate. Tournament selection based on Pareto dominance was first introduced by Horn and Nafpliotis (1993) for multi-objective optimization with genetic algorithms. Kotanchek et al. (2006) presented Pareto tournaments for symbolic regression via genetic programming for both classical GP and Pareto GP. Pareto tournaments are a simple methodology to make classical fitnes-oriented GP Pareto-aware, and to also enhance selection in Pareto GP (compared with selection by single objective tournaments).

It is important to note that all subtrees in a tree representation of an expression are considered as separate models during the selection process. This allows us to effectively cover a lot more search space by a population of a small number of individuals (the number of additional individuals considered together with the host individual is equal to the number of nodes in the tree-representation of the host individual). On the other hand, it also requires

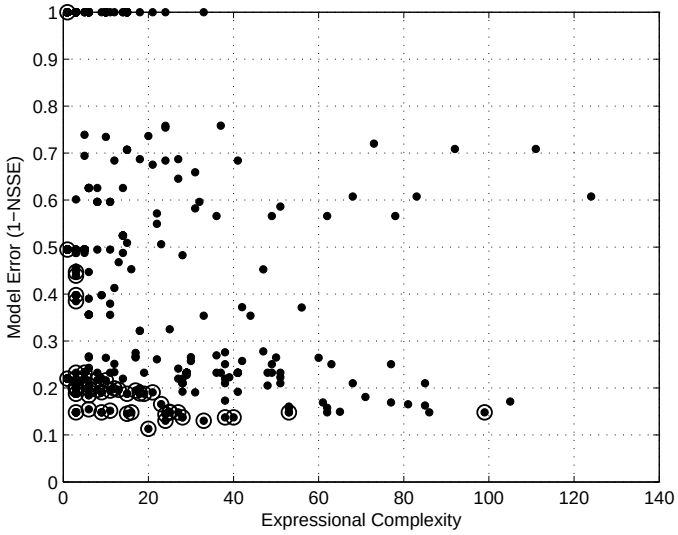


Figure 3.6: Archive at the first generation. The figure illustrates the creation of an archive from the initial population. Dots represent 100 population individuals, which are plotted in the objective space of model expressional complexity and model error ($1 - NMSE$) (all subtrees act as individual models to increase the effectiveness of the search). Fifty individual points from the plotted set, which are dominated by the least number of other individuals, are selected to form an archive (depicted as circles). These archive models will be granted the propagation rights to form the next generation.

the population to have a small number of individuals, since the computational complexity of the non-dominated sorting algorithm is $O(n)^2$, where n is the number of individuals, which is this case becomes bounded by the total number of nodes in the population individuals (duplicate subexpressions, of which there can be many, have to be treated specially). To overcome the problem of large populations, various tricks can be used to reduce the set of sub-expressions to a reasonable subset before applying a non-dominated sorting algorithm (for instance, by excluding unquestionably bad (i.e., dominated) expressions from consideration).

3.2.5 Evolution performance

The goal of Part II of this thesis is to understand and to further enhance the Pareto genetic programming system described in this chapter. Many conclusions on model development are empirical in nature. The next chapters of Part II are structured in such a way that firstly a problem observed in Pareto GP is presented, secondly a new strategy developed to fix this problem is introduced, and finally the strategy is tested on several case studies and compared with the reference Pareto GP[5].

All evolutionary strategies are replicated for 50 to 100 times per case study to obtain statistical significance of results. To compare the performance of the evolutions two main measures are used:

1. Prediction error of archive solutions on the test (unobserved) samples;

2. Area under Pareto front of archive solutions plotted in the objective space of expressional complexity and accuracy.

To assess prediction accuracy the root of the mean squared error on the test data is used. In Chapters 5-7 the independent samples of the best error of the archive are compared for different evolutionary strategies. Chapter 4 studies the numerical stability of archive solutions, and for this reason independent samples of the errors of all archive solutions over independent runs are compared for different evolutionary strategies.

An archive-based GP system does not produce a single best solution, but an archive of solutions. That is why comparing the average or the median best errors on the test set is not sufficient to draw conclusions on the differences in evolution performance; i.e., overall accuracies and complexities of archive solutions have to be compared. To include the archive performance into consideration we compute the average areas under Pareto fronts of archive individuals at the last generation. As areas under Pareto fronts we consider the normalized areas under the convex hulls of the objective vectors corresponding to archive solutions. Note that this is an approximation, convenient for implementation. The Pareto front it not

[5]In fact, we can say that chapters 4-7 are introducing new genetic programming systems, but since representation of individuals is the same, model selection is always bi-objective and is based on non-dominated sorting, and principles are generic, the new approaches are better seen as different evolutionary strategies for Pareto GP.

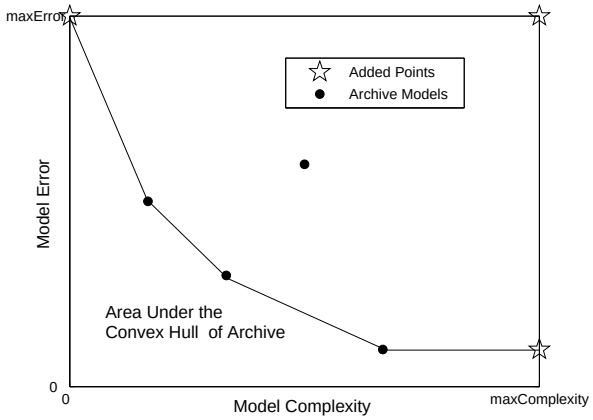


Figure 3.7: Area under the convex hull of a Pareto GP archive.

necessarily convex. Further on, by the area under Pareto front we will mean the area under the convex hull of the Pareto front.

The area under the Pareto front is an interesting measure, since it distinguishes between the runs attaining similar best prediction error but different Pareto front distributions. The smaller the area under Pareto front, the closer the 'knee' of the Pareto front is to zero, and hence, the better opportunities appear to satisfy the customer with solutions that are both sufficiently simple and sufficiently accurate.

To define the area under Pareto front of a set of archive solutions, the following procedure is used:

1. The archive at the last generation is plotted in two objective spaces: expressional complexity versus model error on the training set.

2. The following points are added to the set of objective vectors: $(1, maxError)$[6], $(maxComplexity, maxError)$, $(maxComplexity, minError_{CurrentArchive})$. These points are needed to determine the convex hull of the objective vectors (see Figure 3.7). $MaxComplexity$ is an allowed limit of model complexity. For expressional complexity this limit is usually set to 400 or 500.

[6]There is always a solution with minimum complexity equal to one, e.g. a single node sub-tree labeled with an input variable.

3. An area C of the convex hull of the resulting set of points is computed, and the area under the convex hall is normalized by the total area of the objective space:

$$\frac{maxComplexity \times maxError - C}{maxComplexity \times maxError} 100\%.$$

3.2.6 Implementation details

All simulations of the reference Pareto GP system are performed in the ParetoGP MatLab toolbox developed by Smits (2001) with modifications presented in (Vladislavleva, 2005) and in this chapter. The efficiency of MatLab in handling matrix computations is heavily exploited in the current implementation of Pareto GP, especially for model evaluation and selection.

All computations are performed in double precision.

4

Model Selection through Non-linearity Control

This chapter introduces two new selection strategies for generating symbolic regression models that not only give reliable predictions of the observed data but also have smoother response surfaces and additional generalization capabilities with respect to extrapolation. These models are obtained as solutions of a genetic programming process, where selection is guided by a tradeoff between two competing objectives - prediction accuracy and the order of non-linearity. The latter is a novel complexity measure that adopts the notion of the minimal degree of the best-fit polynomial, approximating an analytical function with a certain precision.

Selection with control over the order of non-linearity is compared with the standard Pareto GP selection with control over the expressional complexity. The latter is shown to have over-fitting problems expressed by pathological behavior of solutions under extrapolation. This result confirms the controversy about 'simplicity that does not always imply generality'. Optimization of the order of non-linearity strongly outperforms 'conventional' optimization of the size-related expressional complexity with respect to extrapolative capabilities of solutions on nine test problems. This suggests that simplicity expressed by the order of non-linearity does imply generality, as opposed to expressional simplicity.

In addition to the exploitation of the new complexity measure, the chapter also introduces a novel heuristic of alternating several (similar) optimization

objectives in a two-dimensional optimization framework. Alternating the objectives at each generation allows us to exploit the effectiveness of two-dimensional optimization when more than two objectives are of interest (in this Chapter, these are prediction accuracy, expressional complexity and the order of non-linearity).

4.1 Motivation

Predictive capability of regression models is mentioned as the #1 requirement to a good model. If nothing at all is known about the behavior of the response surface on the boundary and outside the training range, the only way to improve predictive capability is to produce smoother models with minimized vulnerability within the training range.

This raises a question: How can we measure and control the behavioral complexity of models? And how can we quickly assess their dissimilarities in flexibility to fit the data and in ability to predict the response in a new region?

For the complexity definition, one can think of two directions in determining the qualitative complexity of the GP model: *complexity of the model expression* (compactness or simplicity of the genom) and *behavior of the associated response surface* (smoothness of the phenotype). Since the complexity will have to be determined for all GP individuals on the fly during the GP run, an analytical study of the behavior of a multi-dimensional response surface associated with each individual does not seem feasible. Instead, we are settling for the goal to find a good complexity measure that can be directly computed from a tree structure of the GP individual and still produce high values if the non-linearity of the associated response surface is high.

This chapter aims to explore the feasibility of producing smooth and accurate GP solutions that do not necessarily have 'simple' structures, but have 'simple' response surfaces, and, hence, generalize better. If producing smooth and accurate solutions proves feasible, this will indicate the further possibility of combining the structural and behavioral complexity measures for creating both smooth and simple GP models, without the risk of either *bloat* or *over-fitting*.

There are several structural properties that may reflect the behavior of the function defined by a labeled binary tree:

1. The number of variables in the tree representation (the total number of

variables present at the leaves is an indicator of model complexity; the number of unique variables reflects the dimensionality of the model);

2. The number of binary and unary functions present at inner nodes;

3. Some component-wise non-linearity of functions present at inner nodes (e.g., addition is less nonlinear, and, hence, simpler than exponentiation).

Our main objective in measuring the non-linearity of a model is to favor smooth and extrapolative behavior of the response surface and to discourage highly nonlinear behavior (which is unstable in case of minor changes in inputs and is dangerous for extrapolation).

It would, moreover, be desirable to find a non-linearity measure that agrees with an intuitive impression of the complexity of an elementary function. In other words, we want exponentiation to be more complex than taking a square, summation to be simpler than multiplication, and taking a square root and division to be very complex in the neighborhood of zero.

The first complexity measure aimed to reflect the order of non-linearity of a model was introduced in (Garshina and Vladislavleva, 2004). It is a quantitative measure reflecting the nonlinear growth of the response function determined by a labeled tree. The definition is based on the minimum degree of a polynomial approximating the function on a certain interval with a certain precision.

The reasoning was very simple: An obvious measure for the complexity of a multivariate polynomial is its degree. Any tabulated multivariate function can be associated with its unique *best-fit* approximating polynomial. The degree of this polynomial can be considered as a measure for the order of non-linearity of the response surface of the original function. This order can be seen as a value of the deviation of the response surface from a linear hyperplane.

To make this idea suitable for an evolutionary optimization framework, and in particular for Pareto GP, several severe simplifications had to be introduced.

The best-fit polynomials are difficult to find (see e.g. (Press et al., 1992)). Even though we settle for a polynomial giving a *good* approximation of the function, instead of the best-fit polynomial, we still need a procedure that is computationally efficient. Since our goal is to compare the behavior of response surfaces of GP models at each step of the evolution, we strove to

1. calculate the order of non-linearity iteratively for a given model, starting

from the terminals[1];

2. consider an *efficient* Chebyshev[2] polynomial approximation of a function (Rivlin (1974); Tchebycheff (1857)), instead of a labor-intensive search for an optimal best-fit polynomial for a given precision;

3. determine the order of non-linearity as the degree of the approximating polynomial only for univariate operators, and use another definition of non-linearity for bivariate functions and compositions.

The first implementation briefly described in (Garshina and Vladislavleva, 2004) used a least-squares polynomial approximation for complexity determination. Given a set of points $(x_1, y_1), \ldots, (x_n, y_n)$ and a maximum order p, $(p < n)$, least-squares fitting produces a polynomial $P_{LS} = \sum_{k=0}^{p} a_k x^k$ of degree p that minimizes the error $E_{LS} = \sum_{i=1}^{n} (y_i - \sum_k a_k x_i^k)^2$ with respect to the coefficients a_k. However, certain conditions for points $x_1, \ldots, x_p$ must hold for a least-squares polynomial to exist and to be unique. For high degrees, the problem of finding a unique polynomial often becomes ill-defined. For better treatment of steep response surfaces, we therefore made use of a more stable and reliable framework – Chebyshev approximations (see also (Vladislavleva, 2005)).

The current implementation for a given accuracy constructs a Chebyshev polynomial approximation of a function, and takes the degree of the resulting polynomial as a basis for the measure of non-linearity of a univariate function. An exact definition of the order of non-linearity appears in the next subsection.

Before giving the definition of the complexity measure, we would first like to comment on "approximation of a given accuracy". It is said that $P(x)$ approximates a continuous function $f(x)$ on interval $[a, b]$ with accuracy ϵ, if

$$\max_{x \in [a,b]} |f(x) - P(x)| \leq \epsilon. \tag{4.1}$$

We change the above definition for error evaluation and consider a finite number of samples $x \in S \subset [a, b]$. The way in which we determine the test set S indeed affects the true quality of the approximation. If S has too few points, then condition (4.1) is too weak.

[1] Since we consider the subexpressions of a symbolic model as independent models, we want the order of non-linearity to be easily computable for all subtrees as well as for the parent tree.

[2] Latin spelling of the last name of Pafnuty Lvovich Chebyshev differs in various publications. It is sometimes spelled as 'Tschebyshev'.

If the test set consists of too many points, then error estimation can require excessive computational resources. The choice of S is dictated by a trade-off between the efficiency of computation and the desired accuracy. In the current implementation, the test set S consists of equidistant points whose number changes dynamically depending on the length of the interval $[a, b]$.

Further on, by a polynomial approximation of a univariate function f on the interval $[a, b]$ with a certain precision ϵ, we will denote an approximation in a class of Chebyshev polynomials; more precisely, we define a polynomial $P_f(x) = \sum_{i=0}^{n-1} c_i T_i(x; a, b)$ of a minimal order such that

$$\max_{x \in S \subset [a,b]} |f(x) - P_f(x)| \leq \epsilon, \tag{4.2}$$

where $T_i(x; a, b) = T_i(\frac{2x-(b+a)}{b-a})$, $i = 1, \ldots, n-1$, and $T_i(z)$ is the i-th Chebyshev polynomial on $[-1, 1]$. Details on efficient implementations of Chebyshev polynomial approximations are given in the Appendix of the thesis.

For a univariate function given analytically the degree of the approximating polynomial depends on the interval in which the function is being approximated. This implies the necessity of including scaling into the definition of complexity and calculating the ranges for every inner node that corresponds to a univariate operation. In order to be able to treat bivariate functions as univariate (for polynomial approximations), we also need to estimate the ranges of inner nodes corresponding to bivariate operations.

We would like to emphasize that the function ranges corresponding to inner nodes cannot be determined accurately from the ranges of terminals by using simple interval computations. In general, the range evaluation of a function defined on a real interval should take into account the monotonicity and extrema of this function, and may involve unwanted computation time.

There is a price to pay, however, to avoid these computations. Every subexpression of a symbolic model is explicitly evaluated to obtain the goodness of fit of the predicted output relative to the original output. Therefore, the ranges of subexpressions can be estimated by simply taking the minimum and maximum of the predicted outputs in the fitness evaluation routine. Such evaluation of ranges of all subfunctions of the given model can actually introduce some inaccuracy if the extrema of subfunctions are not at the sampling points. Since this is the best we can do without doing extra calculations, it will be good enough

for an efficient comparison of the non-linearity of the GP models.

Once the ranges for all nodes in the tree-based symbolic model are found, the order of non-linearity of this model can be computed according to the following inductive definition.

4.2 Definition of the order on non-linearity of an expression

Let a tree structure represent a valid analytical model over a set of variables $\mathcal{V} = \{x_1, x_2, \ldots, x_{var}\}$, and a set of constants $\mathcal{C} \subset \mathbf{R}$ with functions from a set $\Phi = \Phi_1 \cup \Phi_2$, where $\Phi_1 = \{sqrt(x), ln(x), exp(x), exp(-x), \sin(x), \cos(x), x^{const}, shift(x), scale(x), const^x\}$, $\Phi_2 = \{x + y,\ x \cdot y,\ x/y,\ x^y\}$. Assuming that the precision ϵ is given, the complexity of the tree structure is calculated from the leaves to the root, according to the following definition:

(A) The complexity of a single node referring to a constant $const \in \mathcal{C}$ is zero:

$$comp(const) = 0. \tag{4.3}$$

(B) The complexity of a single node referring to a variable from $x_i \in \mathcal{V}$ is one:

$$comp(x_i) = 1. \tag{4.4}$$

(C) The complexity of an inner node referring to unary functions $shift(x)$ or $scale(x)$ is equal to the complexity of the child node.

(D) The complexity of an inner node referring to unary function $f \in \Phi_1$, is related to the complexity of the child node referring to a function, variable, or constant, denoted as $g \in \Phi \bigcup \mathcal{V} \bigcup \mathcal{C}$ and the range of the child node $[a, b]$ by the following formula:

$$comp(f \circ g) = comp(g) \cdot n_f, \tag{4.5}$$

where n_f is the minimal degree of P_f, a Chebyshev approximation of the

function $f(x)$, $x \in [a, b]$ with approximation error (4.2) at most ϵ.

Note that the complexity of an inner node referring to a unary function $f \in \Phi_1, \quad f(x, const) = const^x \equiv e^{x \cdot \ln cons}$ is related to the complexity of the child node referring to a function, or to a variable, denoted as $g \in \Phi \bigcup \mathcal{V}$ and the range of the child node $[a, b]$ by the following formula:

$$comp(f \circ g) = comp(e^{g \ln const}) = comp(g) \cdot n_{power}, \tag{4.6}$$

where n_{power} is the minimal degree of a Chebyshev approximation of function $exp^{x \ln const}$, $x \in [a, b]$ with the approximation error (4.2) at most ϵ.

(E) The complexity of an inner node referring to summation and subtraction $\{+, -\} \in \Phi_2$ is related to the complexities of child nodes referring to g_1, g_2 $\in \Phi \bigcup \mathcal{V} \bigcup \mathcal{C}$ by the formulae:

$$comp(g_1 + g_2) = \max\{comp(g_1), comp(g_2)\}, \tag{4.7}$$

$$comp(g_1 - g_2) = \max\{comp(g_1), comp(g_2)\}. \tag{4.8}$$

(F) The complexity of an inner node referring to multiplication $\{*\} \in \Phi_2$ is related to the complexities of child nodes referring to g_1, g_2 from $\Phi \bigcup \mathcal{V} \bigcup \mathcal{C}$ by the formula:

$$comp(g_1 \cdot g_2) = comp(g_1) + comp(g_2). \tag{4.9}$$

(G) The complexity of an inner node referring to division $\{/\} \in \Phi_2$ is related to the complexities of child nodes referring to g_1, g_2 from $\Phi \bigcup \mathcal{V} \bigcup \mathcal{C}$ with ranges $[a, b]$ and $[c, d]$ by the formula:

$$comp(g_1/g_2) = comp(g_1) + comp(g_2) \cdot n_{div}, \tag{4.10}$$

where n_{div} is a minimal degree of the Chebyshev approximation of a function $1/x$ on interval $x \in [c, d]$ with the approximation error (4.2) at most ϵ.

(H) The complexity of the root node determines the complexity of the tree

structure.

The inductive definition described above is an algorithm to calculate the order of non-linearity.

In the current implementation, the maximum admissible degree of the Chebyshev approximation is limited to 100. If the precision of the approximation by an 100-order polynomial still exceeds ϵ, then the complexity value is set to a predefined limit. This limit value is $10,000$ for the current implementation. The value for precision ϵ is fixed to 0.0001. The fixed precision used for estimating the degree of the Chebyshev approximation will imply higher degrees for wider domain ranges. This penalizes the GP system for constructing solutions with too much diversity in the ranges of the inner nodes.

The number of points at which the approximation error is evaluated is dynamic, and depends on the length of the interval of approximation. Currently, we take $\max\{20 \cdot \lceil b-a \rceil, 500\}$ equidistant points on $[a,b]$. This number should vary, depending on the problem difficulty and the descriptiveness of the input data file. An example of the non-linearity calculation for a simple two-variable model with $\epsilon = 10^{-6}$ is given in Figure 4.1.

The definition of the order of non-linearity implies that the complexity of a parent model is never less than the non-linearity of any of its submodels. This definition allows us to implicitly take the complexity of the representation into account, and make the order of non-linearity a characteristic of a **genotype**. Often this causes over-estimation of the true order of non-linearity of the simplified expression. We do this deliberately to push the system towards creating simplified expressions, and to penalize possible precision errors caused by unnecessary scaling. For example, let the trees T_1, T_2, T_3 represent the models x^2/x^2, x/x, and 1 (see Figure 4.2). Despite the fact that the models have identical response surfaces, computation of the values of T_1 may cause loss of precision. This is why the non-linearity complexities of T_1 through T_3 strictly decrease: $comp(T_1) = 12$, $comp(T_2) = 6$, $comp(T_3) = 0$.

A situation may arise in which, during the calculation of the order of non-linearity, we encounter a node corresponding to a function with a very small range (e.g. smaller than 10^{-15}). In this case, we do not evaluate the order of non-linearity by rules (A)-(F), but assign the complexity of the child node to the node complexity. Although the function with a range less than 10^{-15} could be considered as a constant (partly because the accurate approximation of this

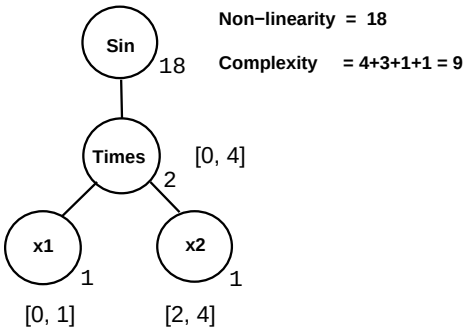


Figure 4.1: Example of non-linearity calculation for a two-variable model. If $x_1 \in [0, 1]$ and $x_2 \in [2, 4]$, then "x_1 *Times* x_2" takes values from the interval $[0, 4]$. Non-linearities of the terminal nodes are one. The non-linearity of the *Times* node is $1 + 1 = 2$. Therefore, the non-linearity of the *Sin* node is two times the degree of the Chebyshev approximation of function $\sin x$ on the interval $[0, 4]$. If the chosen approximation accuracy is 10^{-6}, then the order of non-linearity of the root node is $2 \cdot 9 = 18$. The expressional complexity (visitation length) of the model is 9 (marked as Complexity in the Figure).

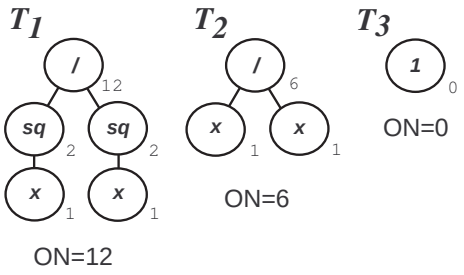


Figure 4.2: Genotypic nature of the order of non-linearity. The order of non-linearity depends on the representation, since interval arithmetics is taken into account: Trees T_1,T_2, T_3 determine models x^2/x^2, x/x, 1, for $x \in [1, 2]$. Whereas the response surfaces of the models are identical, the orders of non-linearity are different.

function and the error evaluation become questionable due to round-off errors), we do not make the order of non-linearity zero, but include the non-linearity of the child tree into the definition.

Since the order of non-linearity is determined inductively for every node of the tree starting from the leaves, an overestimation of the true minimal order of a polynomial approximating the function with given precision will be produced for unary functions in most cases. For example, consider the function $f(x) = sin(x)$ on $[0, 4]$. The minimal degree of the Chebyshev approximation of accuracy 10^{-6} is $n = 9$ and the order of non-linearity of the model is also 9 (see Figures 4.3, 4.4). Now consider function $f(x) = sin(exp(x))$ on $[0, 4]$. The Chebyshev approximation of degree $n = 76$ has accuracy 10^{-6}, but the order of non-linearity of the tree, representing $sin(exp(x))$, is 430 (see Figures 4.3–4.4).

Such overestimation of the true degree of a polynomial approximating a univariate function with a given precision is deliberate. The order of non-linearity represents a relative comparative measure, and also builds a modularity (structural separation) in a space of all possible symbolic models over a given set of variables.

Before turning to the empirical analysis of features of the defined complexity measure, we make one more remark. The problem of constructing a polynomial approximation in $\mathbf{R}^k$ quickly becomes numerically intensive with $k >> 2$. This explains why, in our inductive definition, we treat symbolic models as univariate functions. In general, formulae (4.7), (4.9), (4.10) should not be considered as the rules for finding the true minimal degree of the Chebyshev polynomial.

4.3 Comparisons and case studies

4.3.1 Test problems

Real-life applications operate with complex processes where the true dependency between system inputs and outputs is usually unknown or very complex, and cannot be expressed in one equation. To demonstrate the order of non-linearity control as a mechanism for preventing over-fitting, we selected a suit of synthetic regression problems, which allowed us to generate reliable noise-free test data for interpolation and extrapolation. The target equations for chosen problems are

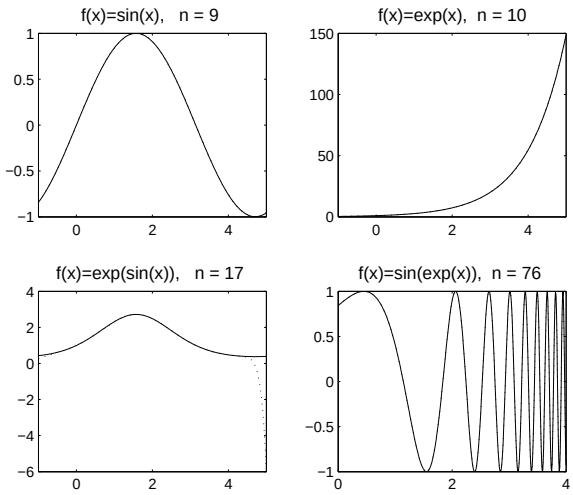


Figure 4.3: Chebyshev approximation with precision 10^{-6} of functions $sin(x)$, $exp(x)$, $exp(sin(x))$, $sin(exp(x))$ on interval $[0, 4]$ and their behavior in extrapolation. The degree n of the approximation is given in plot captions. The orders of non-linearity computed for $x \in [0, 4]$ are shown in Figure 4.4.

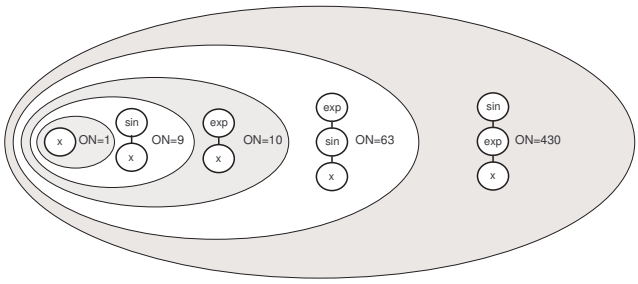


Figure 4.4: Modularity over a set of symbolic models. Orders of non-linearity of tree structures are computed for the domain $x \in [0, 4]$.

given below:

$$f_1(x_1, x_2) = \frac{e^{-(x_1-1)^2}}{1.2 + (x_2 - 2.5)^2} \tag{4.11}$$

$$f_2(x) = e^{-x} x^3 \cos x \sin x (\cos x \sin^2 x - 1) \tag{4.12}$$

$$f_3(x_1, x_2) = f_2(x_1)(x_2 - 5) \tag{4.13}$$

$$f_4(x_1, x_2, x_3, x_4, x_5) = \frac{10}{5 + \sum_{i=1}^{5} (x_i - 3)^2} \tag{4.14}$$

$$f_5(x_1, x_2, x_3) = 30 \frac{(x_1 - 1)(x_3 - 1)}{x_2^2 (x_1 - 10)} \tag{4.15}$$

$$f_6(x_1, x_2) = 6 \sin x_1 \cos x_2 \tag{4.16}$$

$$f_7(x_1, x_2) = (x_1 - 3)(x_2 - 3) + 2 \sin((x_1 - 4)(x_2 - 4)) \tag{4.17}$$

$$f_8(x_1, x_2) = \frac{(x_1 - 3)^4 + (x_2 - 3)^3 - (x_2 - 3)}{(x_2 - 2)^4 + 10} \tag{4.18}$$

The choice of several target expressions was inspired by the paper (Keijzer, 2003), which introduced thirteen functions to analyze the performance of scaled GP. We selected the most difficult problems of that set, and still modified most of them to make the regression process more challenging for our ParetoGP system. The first equation defines the **Kotanchek** function first used in Smits and Kotanchek (2004). The second function originates from Salustowicz and Schmidhuber (1997); we call it the **Salustowicz** function. The third equation (4.13) is our two-dimensional version of the Salustowicz function, which we call **Salustowicz2D**. The function defined in (4.14) is our favorite problem. This five-dimensional equation, which we call the **UBall5D**[3] function, was inspired by a simpler two-dimensional problem from Keijzer (2003); Topchy and Punch (2001). Despite having a simple and harmonious underlying relationship, it appears to be quite difficult for GP. Target expressions for **RatPol3D** (4.15), **SineCosine** (4.16), and **Ripple** (4.17) problems are adopted from Topchy and Punch (2001), with a linear transformation of variables: $x_i \mapsto x_i - 3$, and a few other modifications. The **RatPol2D** problem, defined by equation (4.18), represents another rational polynomial that is challenging for GP. The contour plots of these eight target functions (or projections onto 2D intervals for functions

[3] five-dimensional unwrapped ball

(4.14) and (4.15)) shown in Figures 4.5 and 4.6 illustrate the non-linearity of the underlying response surfaces within the training regions and extrapolation regions.

The data for the ninth test problem come from an industrial problem and represent a gas chromatography measurement of the composition of a distillation tower. This **Tower** problem contains 5000 records and 23 potential input variables.

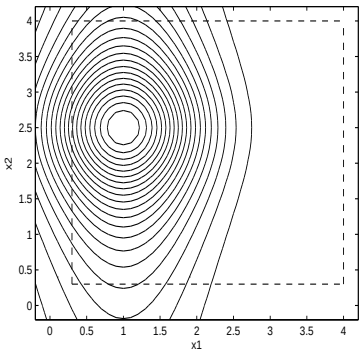
4.3.2 Experimental setup

To compare the effects of optimizing different complexity measures in symbolic regression via GP, we devised experiments for three multi-objective optimization schemes:

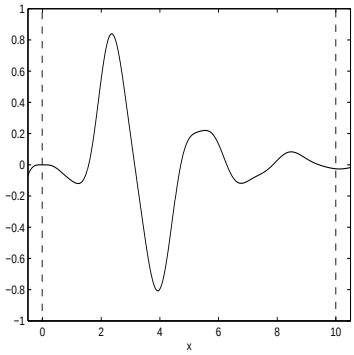
CASE I: Pareto-optimization of the sum of squared errors and expressional complexity;
CASE II: Pareto-optimization of the sum of squared errors and the order of non-linearity;
CASE III: Pareto-optimization of the sum of squared errors and expressional complexity, alternated with the Pareto-optimization of the sum of squared errors and the order of non-linearity *at every generation*.

Note that the optimization of the goodness of fit is present in all cases, since constructing accurate models is our first priority. Comparison of CASE I with CASE II will show that creating accurate and 'structurally simple' (i.e. more compact) equations may still lead to highly nonlinear pathological predictions, compared with creating accurate equations of a low or reduced order of non-linearity.

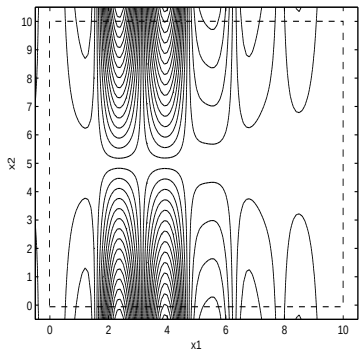
The presence of CASE III experiments is an attempt to blend optimization of the structural complexity and the non-linearity with accuracy optimization in a 'multi-objective' fashion. We do not use a composite objective function, which uses a linear combination of objectives of interest, because of its sensitivity to the particular linear coefficients. Instead of limiting the search by using a composite objective function, one should pursue a true multi-objective search and use a vector of objective functions, whose components are optimized either individually



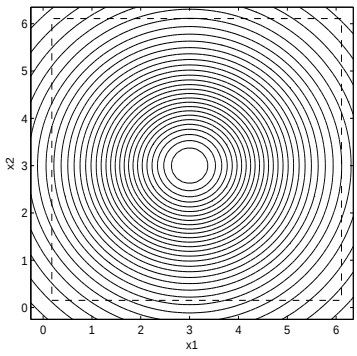
(a) Eq. (5.5) : $\frac{e^{-(x_1-1)^2}}{1.2+(x_2-2.5)^2}$



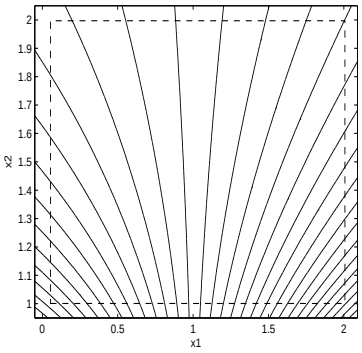
(b) Eq. (4.12)



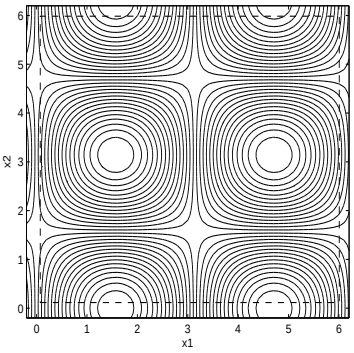
(c) Eq. (4.13)



(d) Eq. (4.14), $x_3 = x_4 = x_5 = 0$

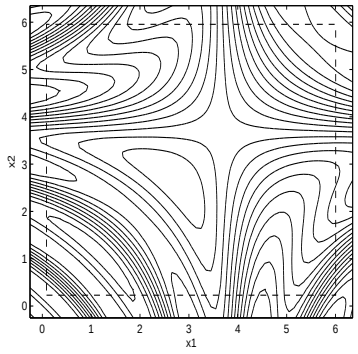


(e) Eq. (4.15) : $30\frac{(x_1-1)}{x_2^2(x_1-10)}$, $x_3 = 2$

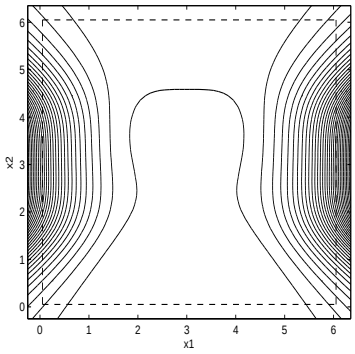


(f) Eq. (4.16) : $6 \sin x_1 \cos x_2$

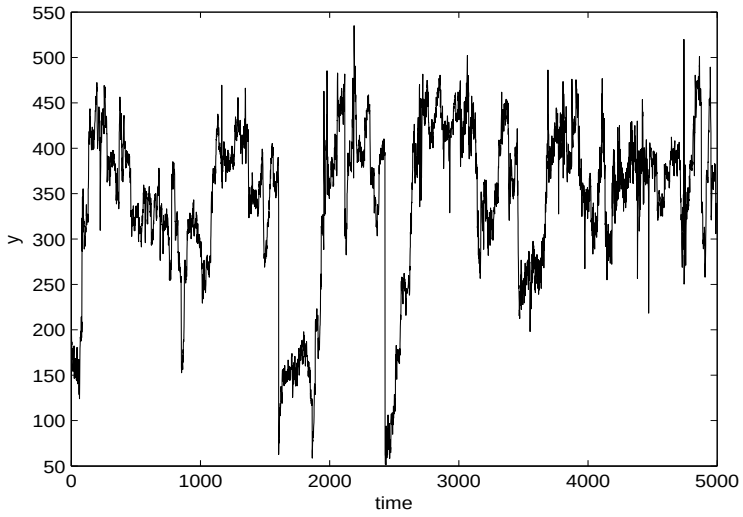
Figure 4.5: Contour plots of the target functions (or their projections) in the test regions. The dashed lines represent the boundaries of the training regions. Note that none of the target response surfaces displays pathological behavior in the test region. We expect the same from the GP solutions with good extrapolation properties.



(a) Eq. (4.17)



(b) Eq. (4.18) : $\frac{(x_1-3)^4+(x_2-3)^3-(x_2-3)}{(x_2-2)^4+10}$



(c) Tower Problem : 5000 response values

Figure 4.6: Contour plots of the target functions (or their projections) in the test regions. The dashed lines represent the boundaries of the training regions. As we see, none of the target response surfaces displays pathological behavior in the test region. We expect the same from the GP solutions with good extrapolation properties.

or simultaneously. However, the multi-objective approach scales badly when the number of objectives increases. We propose a novel heuristic for overcoming the curse of dimensionality of the objective space by alternating between two two-objective optimizations in CASE III.

Since the focus of this Chapter is on non-linearity control, the formalization and generalization of the approach of CASE III for a general multi-objective optimization problem would shift the focus and make the structure complicated. Instead, we illustrate the idea using a particular case in which one accuracy measure and two complexity measures are defined, where the priority is on the accuracy.

Three objectives need to be minimized during the model development cycle: (i) prediction error of the model's phenotype ($Error$ $=1-NMSE$), (ii) the expressional complexity of the model's genom ($Complexity$), and (iii) the order of non-linearity of the model's genotype ($Non-linearity$). The order of importance of these objectives is as follows: error minimization is the primary objective, expressional complexity and non-linearity are equally important secondary objectives. Taking this into account, the original multi-objective optimization problem can be replaced by a different one:

$$\min_{all\ \ generations}(Error,\ \ Complexity,\ \ Non-linearity) \tag{4.19}$$

by

$$\begin{cases} \min\limits_{odd\ \ generations}(Error, \qquad Complexity) \\ \min\limits_{even\ \ generations}(Error,\ \ Non-linearity). \end{cases} \tag{4.20}$$

Computational complexity of the non-dominated sorting algorithm with such substitution drops down from $O(n^3)$ to $O(n^2)$, where n is the total number of models in the population and the archive[4].

The experiments of CASE III were formulated to combine the best properties of the solutions of CASE I and CASE II. Accurate models will thus be produced that are both compact and 'smooth' (i.e. generalize well), while not producing pathologies in the new areas of the input space.

[4]The non-dominated sorting selects a fixed number of models **at** the Pareto front to update the archive.

The results of the experiments will be compared with respect to the number of pathologies produced on test data with extrapolation, average order of non-linearity and expressional complexity and the area percentages under the convex hulls of archives plotted in expressional complexity versus model error and the order of non-linearity versus model error spaces.

The detailed results of CASES I-III follow immediately after descriptions of the settings for GP parameters and the choice of training and test data.

4.3.3 Data sampling and GP settings

As mentioned above, we focus on the synthetic data sets to generate a sufficient amount of reliable, outlier-free test samples in the regions outside the training regions. The details of sampling procedures used for generation of training and test data are given in Table 4.1.

The **Tower** problem contains real-life data for which the true input-output relationship is unknown. To assess extrapolative capabilities of GP solutions for the Tower problem, we decided to select significant input variables at a pre-processing step, and then used only those to divide the data into training and test sets. The driving variables identified at initial screening using the fitness inheritance approach (see Smits et al. (2005)) were x_1, x_4, x_6, x_{12} and x_{23}. The 5000 data records corresponding to these inputs were scaled into the five-dimensional cube $[0, 1]^5$. All records belonging to the interval $[0.02, 0.98]^5$ were selected into the training set, and the remaining records formed a test set for extrapolation.

In all experiments, one optimization measure is always the numerical fitness, determined as a normalized mean-squared error between observed response vector $\mathbf{y}$ and the predicted response vector $\hat{f}(\mathbf{x})$; see Eq. (3.7).

For each approach and for each test problem, 50 independent GP runs are conducted. All GP settings except for the optimization complexity (expressional, order of non-linearity, or "both, but alternating") are the same for each test problem. The number of generations is fixed to 250 for all problems except **SineCosine** and **UBall5D** problems. These two had to be modeled over 500 generations in order to get an appropriate goodness of fit. Other parameter settings are given in Table 4.2 (see also Chapter 2 for details on selection and modification operators).

Table 4.1: Sampling strategy for training and test data for nine regression problems. The table represents the sampling strategy, and the number of points used for training and testing GP solutions. Notation $x = Rand(a, b)$ means that the x variable is sampled randomly from an interval $[a, b]$. Notation $x_1 = (a_1 : c_1 : b_1)$, $x_2 = (a_2 : c_2 : b_2)$ determines a uniform mesh with step length (c_1, c_2) on an interval $[a_1, b_1] \times [a_2, b_2]$.

Problem Name	Training Data	Test Data
Kotanchek Eq (5.5)	100 points x_1, x_2= Rand(0.3,4)	2026 points (x_1, x_2)=(-0.2:0.1:4.2)
Salutowicz Eq (4.12)	100 points x=(0.05:0.1:10)	221 points x=(-0.5:0.05:10.5)
Salutowicz2D Eq (4.13)	601 points x_1=(0.05:0.1:10) x_2=(0.05:2:10.05)	2554 points x_1=(-0.5:0.05:10.5) x_2=(-0.5:0.5:10.5)
UBall5D Eq (4.14)	1024 points x_i=Rand(0.05,6.05)	5000 points x_1=Rand(-0.25,6.35)
RatPol3D Eq (4.15)	300 points x_1, x_3=Rand(0.05,2) x_2=Rand(1,2)	2701 points $x_1, x_3 = (-0.05 : 0.15 : 2.1)$ x_2=(0.95:0.1:2.05)
SineCosine Eq (4.16)	30 points x_1, x_2=Rand(0.1,5.9)	961 points $x_1, x_2 = (-0.05 : 0.02 : 6.05)$
Ripple Eq (4.17)	300 points x_1, x_2=Rand(0.05,6.05)	1000 points x_1, x_3=Rand(-0.25,6.35)
RatPol2D Eq (4.18)	50 points x_1, x_2=Rand(0.05,6.05)	1157 points x_1, x_2=(-0.25:0.2:6.35)
Tower	3136 points all input values in $[0.02, 0.98]$	1864 points one of the inputs in $[0, 0.02) \bigcup (0.98, 1]$

Table 4.2: GP Parameters for nine Test Problems

Number of independent runs	50
Total number of generations	250 and (500 for SinCos and UBall5D)
Population size	100
Archive size	50
Population tournament size	7
Archive tournament size	5
Crossover rate	0.95
Mutation rate	0.05
Rate of mutation on terminals	0.3
Basic Function Set	$+$, $-$, $*$, $/$, $square$ x^{real}, $x + real$, $x \cdot real$
Kotanchek, SineCosine, Tower	Basic Set, e^x, e^{-x}
Salustowicz, Salustowicz2D, Ripple	Basic Set, e^x, e^{-x}, $\sin x$, $\cos x$

4.3.4 Results and discussion

The solutions of each independent GP run are stored in an archive that contains 50 expressions. These 50 individuals lie *at* the Pareto front in 'optimization complexity' versus 'model fitness' objective space containing all individuals evaluated during the current GP run. For our purposes, all of these individuals are equally valuable GP solutions. The 'customer', or the domain expert, will have to choose one of these solutions or, better, an ensemble of solutions that satisfies customer needs; see Kotanchek et al. (2007). We therefore combined all archive solutions of independent runs in one ensemble at a post-analysis stage and analyzed the properties of the resulting set of 50runs$\times$50models= 2500 solutions across different cases and different test problems.

Pareto genetic programming aimed at the generation of a multitude of solutions in the archive, is yet a rarity rather than a convention in symbolic regression applications of GP. The most common way to compare performances of different evolutions is the comparison of their best-of-the-run individuals. Empirical analysis of the experiments of this chapter includes the features of the best-of-the-run solutions as well.

The detailed results are given in Part I and Part II of Table 4.3. The columns

(2) and (3) of this table contain the fraction of equations that showed pathological behavior on the test data. Column (2) is the percentage of equations producing infinite or undefined root mean-squared error. The latter is computed from the vectors of predicted values of the model, $\hat{f}(\tilde{\mathbf{x}})$, and the target values on the test data, $\tilde{\mathbf{y}}$, as:

$$RMSE(\tilde{\mathbf{y}}, \hat{f}(\tilde{\mathbf{x}})) = \sqrt{\frac{1}{N_T}\sum_{i=1}^{N_T}(\tilde{y}^i - \hat{f}(\tilde{\mathbf{x}}^\mathbf{i})^2}, \tag{4.21}$$

where N_T is the number of records in the test set. Individual $\hat{f}$ is optimally scaled on the training data, before the errors on the test set are computed (see Chapter 3).

Column (3) of Table 4.3 is the percentage of solutions for which the root mean-squared error is infinite, undefined, or excessively large. The threshold for pathologically high error is chosen to be 100. It corresponds to the mean-squared error equal to 10^4. Equations producing these large errors on test data are dangerously erroneous, and a high fraction of them in the set of solutions indicates the tendency to over-fitting.

Column (4) of Table 4.3 represents the percentage of equations that have the highest allowed order of non-linearity, equal to $10,000$. This value indicates highly non-linear behavior of a model and the potential to have a pathology on new data. We expect best-of-the-run solutions to have these high non-linearity values, due to their inclination to over-fitting. Since the rest of the solutions are expected to have lower non-linearity, small percentages in column (4) are preferred.

Column (5) contains the mean order of non-linearity of those solutions that have the non-linearity below the threshold of $10,000$. Low values in this column for solutions of CASE I for **Salustowicz**, **Salustowicz2D**, **UBall5D**, and **SineCosine** problems demonstrate the 'all or nothing' phenomenon for the orders of non-linearity of models generated by expressional complexity minimization. For example, for CASE I of the Salustowicz problem, we see that 86% of final solutions have the order of non-linearity $10,000$, and the *remaining* 14% have average non-linearity equal to only 12.2. We illustrate such a situation for one GP run in Figure 4.7.

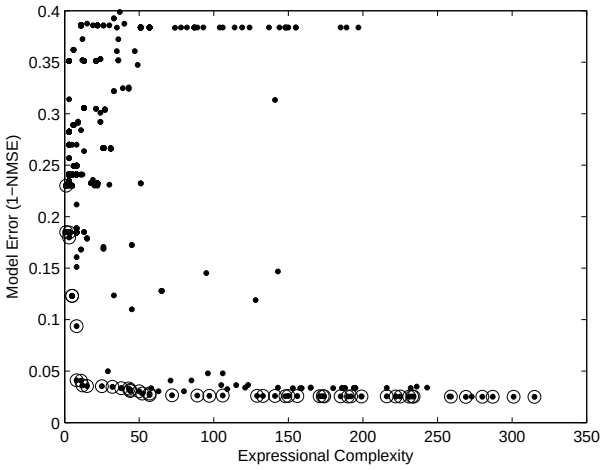
Column (6) contains the average expressional complexity of 2500 solutions among independent runs. From columns (2) to (6) we observe that CASE I

Table 4.3: Part I. Results of the three experiments on selected test problems for ALL solutions. Case I consists of optimization of fitness and expressional complexity, Case II - optimization of fitness and the order of non-linearity, and Case III - optimization of the fitness and alternated at each generation expressional complexity and non-linearity. For each experiment the quality of archive solutions and the best-of-run solutions over 50 independent runs is assessed against training and test data with extrapolation. Case II consistently outperforms other experiments with respect to the rate of pathologies on test data over all test problems.

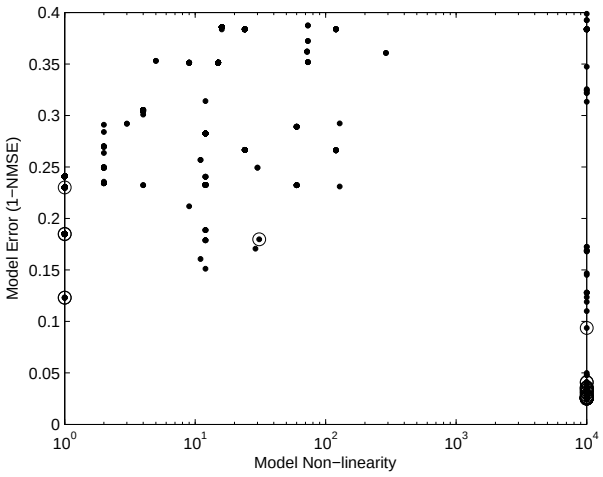
Problem and Experiment	ALL final solutions over all runs (2500)				
	Pathologies on Test Data		Complexity		
			Order of Non-linearity		Expressional
	RMSE $= \infty$, %	RMSE ≥ 100, %	$= 10^4$, %	$< 10^4$, mean	mean
Column Number	(2)	(3)	(4)	(5)	(6)
Kotanchek					
CASE I	29%	30%	41%	509.1	106.2
CASE II	**5**%	7%	2%	327.9	243.3
CASE III	**10**%	10%	6%	383.7	114.9
Salustowicz					
CASE I	34%	36%	86%	12.2	111.9
CASE II	**11**%	28%	31%	38.9	194.6
CASE III	17%	19%	37%	18.7	132.2
Salustowicz2D					
CASE I	36%	47%	82%	29.0	96.8
CASE II	**8**%	20%	5%	134.4	226.1
CASE III	18%	23%	23%	227.7	99.7
UBall5D					
CASE I	43%	45%	92%	46.0	141.2
CASE II	**16**%	18%	4%	33.3	254.5
CASE III	12%	14%	10%	80.6	93.4
RatPol3D					
CASE I	15%	15%	22%	58.7	104.6
CASE II	5%	5%	1%	29.4	262.0
CASE III	10%	10%	3%	31.4	122.7
SineCosine					
CASE I	12%	26%	78%	77.0	132.9
CASE II	8%	17%	8%	96.0	348.0
CASE III	12%	19%	21%	119.2	155.0
Ripple					
CASE I	26%	26%	30%	298.2	99.1
CASE II	6%	7%	2%	114.3	250.4
CASE III	7%	8%	6%	220.5	130.0
RatPol2D					
CASE I	52%	55%	42%	108.9	109.3
CASE II	14%	15%	2%	18.3	241.1
CASE III	23%	24%	4%	32.8	117.9
Tower					
CASE I	19%	19%	1%	38.2	84.8
CASE II	10%	10%	0%	50.0	257.6
CASE III	15%	15%	0%	67.1	112.5

Table 4.3: Part II. Results of the three experiments on selected test problems for BEST-of-the-RUN solutions. Case I consists of optimization of fitness and expressional complexity, Case II - optimization of fitness and the order of non-linearity, and Case III - optimization of the fitness and alternated at each generation expressional complexity and non-linearity. For each experiment the quality of archive solutions and the best-of-run solutions over 50 independent runs is assessed against training and test data with extrapolation. Case II consistently outperforms other experiments with respect to the rate of pathologies on test data over all test problems.

Problem and Experiment	BEST-of-the-RUN solutions over all runs (50)						
	Model Error					Complexity	
	Training Data, RMSE		Test Data, RMSE			Order of Non-lin.	Express.
	median	IQR	$= \infty$, %	≤ 100, median	IQR	mean	mean
Column Number	(7)	(8)	(9)	(10)	(11)	(12)	(13)
Kotanchek							
CASE I	0.052	0.02	52%	0.075	0.06	7050	291
CASE II	0.055	0.03	18%	0.072	0.06	2476	350
CASE III	0.052	0.02	42%	0.069	0.04	5239	319
Salustowicz							
CASE I	0.216	0.07	24%	0.233	0.90	10000	298
CASE II	0.218	0.08	28%	0.212	0.14	9604	351
CASE III	0.212	0.10	34%	0.229	0.50	9804	337
Salustowicz2D							
CASE I	0.938	0.20	50%	1.061	0.52	9802	276
CASE II	0.832	0.54	24%	0.739	0.69	5095	320
CASE III	0.932	0.31	50%	0.817	0.47	9040	293
UBall5D							
CASE I	0.173	0.03	66%	0.277	0.86	10000	329
CASE II	0.183	0.01	42%	0.637	1.16	4314	377
CASE III	0.178	0.02	32%	1.024	1.60	7109	359
RatPol3D							
CASE I	0.221	0.01	24%	1.037	0.05	3885	292
CASE II	0.218	0.02	8%	1.029	0.03	887	357
CASE III	0.218	0.02	22%	1.033	0.04	1838	317
SineCosine							
CASE I	1.393	0.33	22%	17.869	28.89	9294	311
CASE II	1.328	0.28	12%	3.469	7.58	5288	472
CASE III	1.270	0.32	22%	7.599	26.16	9203	449
Ripple							
CASE I	1.325	0.21	42%	1.457	1.29	5870	278
CASE II	1.311	0.10	10%	1.467	0.52	2037	341
CASE III	1.312	0.15	20%	1.486	0.48	3598	308
RatPol2D							
CASE I	0.612	0.34	50%	3.881	8.34	6115	283
CASE II	0.553	0.20	26%	2.063	1.78	1236	367
CASE III	0.663	0.21	38%	3.179	3.34	3263	337
Tower							
CASE I	30.3	1.38	42%	40.4	8.4	741	293
CASE II	30.1	1.43	24%	40.7	7.8	462	349
CASE III	30.1	1.37	32%	42.7	7.5	500	294



(a) Original objective space



(b) Non-linearity vs. Model Error

Figure 4.7: Solutions of a GP run of CASE I experiment for the Salustowicz2D problem plotted in different objective spaces. In the upper plot, solutions of a CASE I (circled dots) run and their sub-equations (black dots) are mapped to the original objective space – Expressional Complexity versus Model Error. We see a horizontal trend in solutions; i.e., most of the equations have similar errors, despite growth in expressional complexity. If we plot the same archive solutions in a different objective space of the order of non-linearity versus model error (bottom plot), we observe a big imbalance in the distribution of the non-linearity. All solutions with errors below 0.1 (45 out of 50) reach the non-linearity threshold. Of the five remaining equations four have the non-linearity equal to one (two very similar errors, and therefore appear as one circled dot in the plot); that is, they are estimated as linear in the original inputs.

produces, on average, more compact expressions than CASE II (see column six), although, a greater fraction of these have pathologies on test data (see columns two and three). We performed pair-wise statistical significance tests for solutions of CASES I-III, and concluded that CASE II outperforms CASE I with respect to the error on test data for all test problems (see Table 4.4, columns two and three). For significance tests, the solutions with infinite errors (or errors higher than 100) were assigned an error value of 100. ANOVA tests and Wilcoxon-Mann-Whitney rank sum tests were performed to compare the means and the medians of different sets of error values of the equal number of samples. Table 4.4 represents the p-values for the 95% significance level. The mean and the median error on test data for the solutions of CASE II were significantly smaller than those of the solutions of CASE I (the maximum p-value is 0.0004 for the Wilcoxon test on the **RatPol3D** problem).

The smoothness of solutions of CASE II comes at the expense of higher expressional complexity. Solutions, generated in CASE II consist, for the most part, of 'simple' operators (such as addition, subtraction, multiplication), but are bulky and sometimes difficult to interpret. This excessive growth in structure disappears when the CASE III experiment is used. Comparing the results of CASE II and CASE III in Table 4.3, we observe that the average expressional complexity of solutions can be reduced in CASE III (column six), however, with a side effect of an increased pathology rate (columns two and three). The significance tests show that CASE II significantly outperforms CASE III in the error on the test set on eight out of nine test problems: Kotanchek, UBall5D (only for the mean error), RatPol3D, RatPol2D, Tower, and SineCosine and Ripple (only for the median error).

In a comparison of CASE III with CASE I, the tests show that the errors on the test data produced by solutions of CASE III are significantly smaller than those of CASE I in all test problems (see columns two and three of Table 4.4). This brings us to the first important conclusion: solutions obtained in CASE III with alternating expressional complexity and the order of non-linearity, as well as solutions of CASE II with non-linearity minimization, are significantly smoother than the ones of CASE I with minimization of expressional complexity. The fact that CASE III produces solutions competitive with CASE I (with respect to the error) and CASE II (with respect to the expressional complexity) is counter-intuitive and quite surprising. The alternation of optimization complexities *at*

each generation is a very crude heuristic aimed at producing both compact and smooth equations. The fact that it works motivates us to explore the scalability of this approach to cases in which a multitude of objectives needs to be satisfied.

The second part of Table 4.3 shows the results for the 50 best-of-the-run solutions for each case. The median and the interquartile range of the root mean-squared error over these 50 best-of-the-run solutions per experiment are given in columns seven and eight of Table 4.3. The resulting values look similar for CASES I-III, and no clear trends can be observed with respect to the superiority of any one approach on the training data.

The differences among solutions of CASES I-III become obvious when the best-of-the-run equations are evaluated on the test sets. Column (9) of Table 4.3 represents the fraction of best-of-the-run equations that have a pathology on the test data, defined as an infinite root mean-squared error. The trend is similar to the one revealed in columns two and three: CASE I has the highest rate of pathologies at extrapolation. The only exception in this rule is the Salustowicz problem, with 24%, 28%, and 34% pathological equations from the 50 best-of-the-run equations.

Columns (10) and (11) of Table 4.3 contain the median error and the interquartile range of a set of errors that are smaller than 100. For example, for the solutions of CASE I of the **Kotanchek** problem we observe that 52% of best-of-the-run solutions (26 equations out of 50) have a pathology on the test data. If those and also other equations producing errors higher than 100 are removed from the sample, then the median of the remaining 24 equations will be 0.075, and the interquartile range will be 0.06. This is an argument for using archives of equations and for being very cautious when using best-of-run solutions. If only best-of-the-run solutions are sought for, then all runs where the best equation produces a pathology are lost. This corresponds to an incredible waste of 52% *of the spent effort* in the example of CASE I solutions of the Kotanchek problem.

The significance tests for the errors of best-of-the-run equations are performed in a style similar to that used for all solutions. We first assign an error value of 100 to equations producing undefined and infinite values, as well as those exceeding 100. We then perform the pair-wise ANOVA and Wilcoxon tests to determine the significance in the difference of the mean and median errors among 50 solutions of CASES I-III. The p-values of the tests are given in columns three and four of Table 4.4. We can observe that CASE II significantly outperforms

Table 4.4: Significance of conclusions about the results of the three experiments. We performed one-way Anova tests and Wilcoxon-Mann-Whitney tests for analyzing differences in the means and the medians of accuracy values of solutions of CASES I-III. The p-values for ANOVA tests are obtained from the F-statistics ant the 95% confidence level (ANOVA). P-values for Wilcoxon-Mann-Whitney tests are obtained using the Z-statistics and the 95% confidence level and are doubled for two-sided tests (Wilcoxon). To obtain equal sample sizes, we truncated all infinite and excessively large values of RMSE (those where RMSE>= 100) on the Test Data at 100.

Problem and Hypothesis	Comparing the RMSE, TEST data all (2500)		best-of-run (50)		Comparing the Average Area %	
	AN., p-value	Wilc., p-value	AN., p-value	Wilc., p-value	AN., p-value	Wilc., p-value
Column Number	(2)	(3)	(4)	(5)	(6)	(7)
Kotanchek						
CASE II outperforms CASE I	0	0	0.0003	0.0057	0.2924	0.2715
CASE II outperforms CASE III	9E-07	0.0000	0.008	0.1936	0.0504	0.0137
CASE III outperforms CASE I	0	6E-33	0.31	0.1962	0.3625	0.2539
Salustowicz						
CASE II outperforms CASE I	0	0	0.3312	0.9579	0	0
CASE III outperforms CASE II	0	0	0.5606	0.8017	0.2696	0.29
CASE III outperforms CASE I	0	0	0.6971	0.8664	0.0000	0.0000
Salustowicz2D						
CASE II outperforms CASE I	0	0	0.0184	0.0006	9E-21	9E-16
CASE II outperforms CASE III	0.0462	0.0055	0.0148	0.0095	0.0361	0.0919
CASE III outperforms CASE I	0	0	0.9295	0.5181	3E-12	4E-11
UBall5D						
CASE II outperforms CASE I	0	0	0.0328	0.1379	1E-14	5E-12
CASE II outperforms CASE III	0	0.8655	0.5058	0.9858	-	-
CASE III outperforms CASE II	-	-	-	-	0.0025	0.0009
CASE III outperforms CASE I	0	0	0.0047	0.0908	1E-12	9E-13
RatPol3D						
CASE II outperforms CASE I	0	0.0004	0.0287	0.0579	-	-
CASE I outperforms CASE II	-	-	-	-	0.0000	0.0000
CASE II outperforms CASE III	0	0	0.0503	0.0824	-	-
CASE III outperforms CASE II	-	-	-	-	3.E-07	1.E-07
CASE III outperforms CASE I	0	0	0.8117	0.8109	0.7237	0.3647
SineCosine						
CASE II outperforms CASE I	0	0	0.0006	0.0000	0.1982	0.1168
CASE II outperforms CASE III	0.1288	0	0.9963	0.2711	0.7107	0.8388
CASE III outperforms CASE I	0	0	0.9847	0.0003	0.0819	0.0682
Ripple						
CASE II outperforms CASE I	0	0	0.0006	0.0109	0.8438	0.39
CASE II outperforms CASE III	0.3902	0	0.1318	0.4248	-	-
CASE III outperforms CASE I	0	0	0.0501	0.0925	-	-
RatPol2D						
CASE II outperforms CASE I	0	0	0.0004	0.0000	0.9015	0.5884
CASE II outperforms CASE III	0	0	0.0305	0.0096	0.5995	0.8985
CASE III outperforms CASE I	0	0	0.1707	0.0728	0.6196	0.9149
Tower						
CASE II outperforms CASE I	0	0	0.08	0.1769	-	-
CASE I outperforms CASE II	-	-	-	-	0.0000	0.0000
CASE II outperforms CASE III	0	0	0.2921	0.1348	-	-
CASE III outperforms CASE II	-	-	-	-	0.0000	0.0000
CASE III outperforms CASE I	0	0	0.5648	0.9661	0.8767	0.7329

CASE I with respect to best-of-the-run errors on test data on seven out of nine problems (for **UBall5D** only for the mean error).

CASE III significantly outperforms CASE I on best-of-the-run errors only for the mean error on the **UBall5D** problem. There is no significant difference in the error samples of CASE III and CASE I for the rest of the problems. The second important conclusion that we can make for CASE III is that it is *nowhere* significantly worse than CASE I – even on best-of-the-run solutions.

The average values of the order of non-linearity and expressional complexity of the best-of-the-run equations are given in columns (12) and (13) of Table 4.3. The conclusions about the complexity of best-of-the-run solutions are the same:

1. CASE I produces more compact expressions at the expense of high orders of non-linearity;
2. CASE II produces solutions with lower non-linearity, but higher expressional complexity;
3. CASE III produces lower orders of non-linearity than CASE I does, and lower expressional complexity than CASE II.

4.3.5 Analysis of the evolved GP models

Note that the differences in the expressional complexity of best-of-the-run equations for CASES I-III are not big. As an example, we give the five best-of-the-run solutions for each CASE on the **Salustowicz** problem in Table 4.5. All equations in Table 4.5 are simplified in Mathematica, so the solutions of CASE II appear to be short (since the linear operations on constants and input variables are already executed). The purpose of Table 4.5 is to provide the reader with a visual impression of the differences primarily between CASE I and CASE II solutions and to support our claim that shorter equations may be less convincing for an engineer than longer but less non-linear equations. Formulae of solutions of CASE I in Table 4.5 illustrate that opting for shorter equations generates many nested functions that may make no physical sense.

4.3.6 Further discussion: areas under Pareto fronts

To conclude the performance analysis of CASES I-III, we compared the average areas under the convex hulls of the archive at the last generation. Since CASES

Table 4.5: Examples of best-of-the-run simplified solutions for the Salustowicz2d problem from five independent runs.

CASE I

1) $$-\sin\left[\sin\left[\frac{x_1}{\frac{1.7082+1.00066e^{-x_1}}{(-1+x_1)^2x_1^2-2.369x_2}-5.355e^{x_1}x_2+(2.369+5.355x_2+\sin[\sin[x_2]])^2}\right]\right]$$

2) $$\frac{\cos\left[x_2+\left(x_2+\frac{\sin[x_1]}{(x_1+\sin[x_1+(6.7169+x_2)^2])^2}\right)^2\right]}{\left(x_1+\frac{x_1^2}{(6.7169+x_1-1.x_1^2)^2}+(-6.7169-2.x_1+x_1^2)^2\right)^2}$$

3) $$\left(-8+x_2^{3.5089}\right)\left(3.8823+7.4638x_1^2+\frac{\left(-15.4146+(-7.7073+x_1)x_1+\frac{8}{x_2}+x_2\right)^2}{\cos[64]+\frac{x_1}{-7.7073-\frac{0.0518828}{\frac{x_1}{-7.7073+\frac{1}{x_1^{1.5467}}}+\cos[64]}}}\right)$$

4) $$5.33706-2.3856\text{Sec}[4.4562-\cos[8.932x_2]]\sin[3.8411+2.0392x_1+\cos[x_2]]+$$
$$\ldots\ (-4.4562+\cos[8.932x_2])\left(-4.4562+\frac{\sin\left[\frac{\sin[e^{x_2}]}{-3.8411+x_1}\right]}{-3.8411+x_1}\right)$$

5) $$\frac{x_1-x_2}{x_1^2-x_2}+x_2\cos\left[\frac{x_1\left(x_1+\frac{x_2}{1-x_1}+x_2\cos\left[\cos\left[\frac{2.4388}{x_1}\right]\right]\right)}{-x_2+\frac{\cos[x_1]}{x_1}}\right]$$

CASE II

1) $$x_1(-x_1^{1.185}+x_1(-8.8+12202.5x_1(-8.8422+(-8.8422+x_1)x_2))+\frac{1}{(4.3648-1.x_1)^2}\times$$
$$\sin[x_1^2])^2\sin[x_2]\ (9.4094-x_1)x_2^2(9.4094+3.3661x_1+x_2)$$

2) $$(x_1-x_2)\left(x_1-1.6784\left(-7.7516+x_1(-3.-7.7516x_2)+x_2+x_1^2x_2\right)\cos[7.7516-2x_1]\right)$$

3) $$e^{-x_2-\cos[x_1]-\cos[x_2]}\cos[1.7922-2x_1]\cos[5.418-\cos[0.7922x_1]]$$

4) $$\left(10.3834-x_1-\sin[x_1]^2\right)(-4.1467++4.8212(-e^{-\sin[x_1]}+4.8212(-4.3834+e^{-2x_1}-$$
$$\cdots-e^{-x_2}+x_2)+\cos[x_1+x_2])\sin[x_1-\sin[x_1]])$$

5) $$x_1(-2.2003-x_1-9.4325(0.466332-9.4325x_1)(-x_1+(-7.8043+x_1)^2(x_1+$$
$$\ldots(-9.9167+x_1)^2(22.7295-4.8807x_2)))\cos[7.8043-x_1]\cos[x_1])$$

CASE III

1) $$-\frac{0.123944(-2.8733+x_1)(-4.9692+x_2)}{\left(x_1-0.035185\left(-2.8733+x_1^{1.7365}-\left(-\frac{7.1755}{x_1}+x_1\right)^4\right)\right)^2}$$

2) $$\frac{1}{-3.+x_1}(-3.5211+x_2)(1.-1.\cos[3.0688-3.0688x_1])\times$$
$$\cdots\times\left(-2.+2.\cos[x_1]-1.\cos\left[2+\frac{6.1376}{-2.+x_2}\right]+\cos[0.6542-\cos[3.0688-3.0688x_1]]\right)$$

3) $$5.8277(-7.7992+x_1)^2x_1^2(5.8277(-7.7992+x_1)(-7.7992+x_1(-5.2613+x_2))+$$
$$\cdots+x_1(-5.2613+x_2)(x_1+5.8277(-7.7992+x_1)\sin[7.7992-x_1])^2)\sin[x_1]$$

4) $$-4.318-\frac{e^{\sin[\sin[5-x_1]]}(x_1+x_2)\left(x_1+2\left(-5+x_2^2\right)\right)\sin[2x_1]}{x_1x_2}+$$
$$\cdots+\sin\left[\frac{x_1}{-11.9479+x_2+x_2^2}\right]$$

5) $$\sin[2x_1]/((x_1+\cos[x_1]+(27.4092x_2+\cos[2x_1-x_2])(1.9861+\sin[x_1]))(x_1+(27.4092x_2+$$
$$\cdots+\cos[x_1-x_2])^2(7.6219+\sin[x_1])))$$

I-III exploit different complexity measures, we use the concept of the area under the Pareto front in the following definition of the performance characteristic of a GP run:

1. The archive at the last generation is plotted in two objective spaces: expressional complexity versus model error, and order of non-linearity versus model error.

2. In both objective spaces the following points are added to the set of solutions: $(1, maxError)$, $(maxComplexity, maxError)$, $(maxComplexity, minError_{CurrentArchive})$. These points are needed to determine the convex hull of the solutions in each objective space (see Figure 3.7).

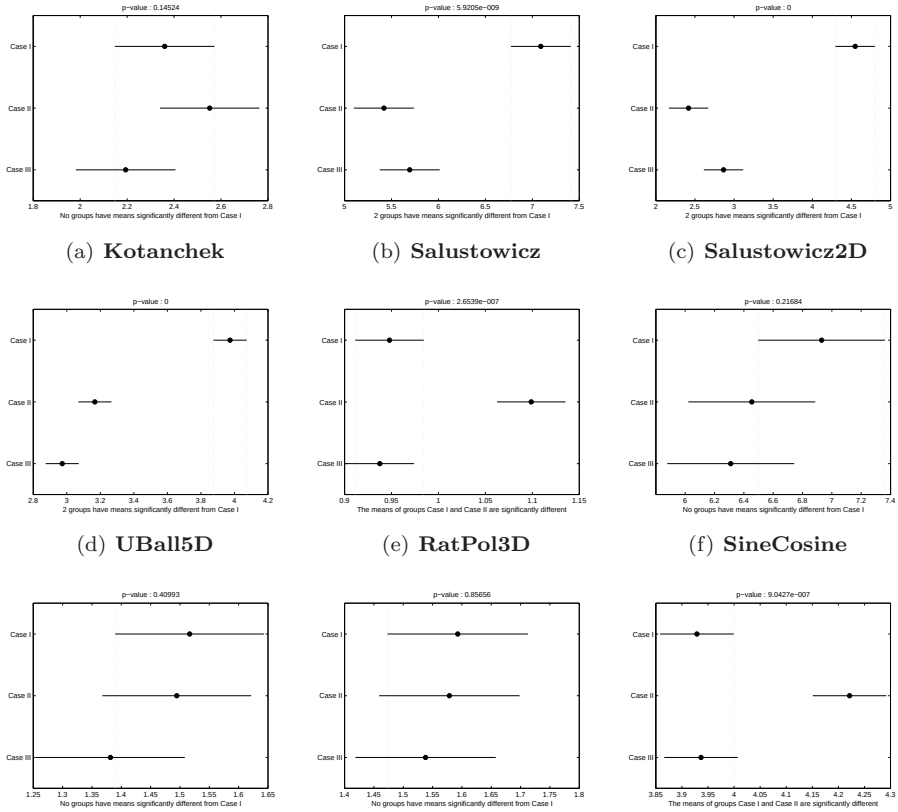
3. An area C of the convex hull of the resulting set of points is computed together with the following percentage:

$$\frac{maxComplexity \times maxError - C}{maxComplexity \times maxError} 100\%.$$

4. The final area percentage for the GP run is defined as an average of the two area percentages under the convex hull of the archive, computed in the expressional complexity versus error and in the order of non-linearity versus error objective spaces.

We computed the average area percentages of two objective spaces for independent runs of each experiment, and performed the multiple comparison tests for significance in the differences of the mean values for CASES I-III (ANOVA tests at the 95% confidence level). The results of these comparisons are plotted in Figure 4.8.

Columns (6) and (7) of Table 4.4 reported the p-values for the pair-wise comparisons of the average area percentage with the ANOVA tests and the Wilcoxon tests. The conclusion from these statistical tests: CASE III is statistically better than CASE I on three of the nine problems (**Salustowicz**, **Salustowicz2D**, **UBall5D**) and is not statistically different from CASE I on the rest of the problems. CASE III is better than CASE II on three problems (**UBall5D**, **RatPol3D**,**Tower**), and is statistically the same for the rest of the problems. These results make CASE III the winner in the comparison for



(a) **Kotanchek** (b) **Salustowicz** (c) **Salustowicz2D**

(d) **UBall5D** (e) **RatPol3D** (f) **SineCosine**

(g) **Ripple** (h) **RatPol2D** (i) **Tower**

Figure 4.8: Multiple comparison tests for average percentage of areas under the convex hulls of archive solutions for nine test problems. Plots represent the 95% confidence intervals of the two-sided tests for multiple comparison of the means of average area percentages over 50 independent runs. Smaller values of the average area percentage are preferred. Labels of the vertical axis of the plots correspond to CASE I, CASE II, and CASE III (read from the top down). Groups are significantly different if the confidence intervals do not overlap. The surprising observation is that CASE I (with expressional complexity minimization) does **not** outperform CASE III (with alternating complexities) on any test problem.

producing the least average percentage area under the convex hull of the archive in two objective spaces.

4.4 Summary

This chapter has introduced a novel complexity measure for creating smoother individuals in symbolic regression via genetic programming. The suggested measure is computed iteratively for *genotypes* of symbolic models according to the set of rules (A)-(G). The notion of the new measure is based on a degree of Chebyshev polynomial approximation of a certain accuracy.

The positive effects of controlling the order of non-linearity are demonstrated for two strategies in nine non-linear test problems.

One of the weaknesses of the order of non-linearity is an over-estimation of the true minimal degree of Chebyshev approximation of accuracy ϵ for unary functions and the approximate nature of the definition for functions of multiple arguments. Even for functions of two arguments, constructing a Chebyshev approximation is performed in terms of tensor products, and represents a non-trivial computational procedure. For functions of more variables it is difficult to construct the Chebyshev polynomial approximation of a given accuracy, thus making it difficult to compare the order of non-linearity with the degree of such an approximation.

The presented order of non-linearity applied as a second optimization objective in combination with numerical accuracy to symbolic regression via Pareto GP favors models with smoother response surfaces. On all nine test problems, these models show significantly better extrapolative capabilities over models generated with controlled expressional complexity.

Models generated with minimization of the order of non-linearity (CASE II experiments) are less compact than those generated via optimization of expressional complexity (CASE I experiments). To combine the benefits of creating compact expressions with smoother response surfaces, we have proposed a new hybrid approach to symbolic regression: Pareto-optimization of the goodness of fit and expressional complexity, alternated with the Pareto-optimization of the goodness of fit and the order of non-linearity at every generation (CASE III experiments).

The vast majority of models obtained with the order of non-linearity

control (CASE II and CASE III) do not get discontinuities when extrapolated over reasonable distances. This agrees with out conjecture that smoother approximations mimic the original output longer when extrapolated outside the training range.

The fundamental question regarding the way in which to obtain a reliable prediction of the output for an extrapolated domain remains a subject for further research.

One definite conclusion of this study is that explicit control of over-fitting is crucial in developing regression models. Even if individuals producing indefinite prediction errors in the training set are removed from populations, the creative power of evolution may lead to a generation of individuals with intrinsic potential for pathologies 'in-between' the training points, *even* if their structural complexity is explicitly controlled. It is essential to mitigate the risk of such pathologies in the unobserved regions; otherwise the models will be inapplicable for deployment. Interval arithmetic is one of the ways of risk reduction. If data is abundant, than the potentially pathological behavior of GP solutions can be reduced, but not avoided, by evaluating their fitness on validation data sets (see e.g. (Gagné et al., 2006)). The potential benefit of explicit non-linearity control presented in this chapter (performed independently from interval arithmetic calculations), lies in the fact that it explicitly maintains the diversity of solutions with respect to non-linearity in addition to mitigating the risk of pathologies among best-of-the-run solutions.

5

Model Evaluation through Data Balancing

Symbolic regression of input-output data conventionally treats data records equally with few exceptions. Chapter 2 of this thesis has introduced a framework for automatic assignment of weights to data samples that takes into account the data's relative importance. This chapter studies the possibilities of improving symbolic regression on imbalanced data by incorporation of weights into the fitness function.

For cases where given input-output data contains some redundancy, the chapter suggests an approach to considerably improve the effectiveness of regression by applying more modelling effort to a smaller subset of given data, which has a similar information content. Such an improvement can be achieved due to better exploration of the search space of potential solutions with the same number of function evaluations.

Different approaches to weighted regression are compared through five test problems with a fixed budget allocation (with budget measured in the number of function evaluations). The results of the case studies demonstrate that a significant improvement in the quality of regression models can be obtained either with weighted regression (when computed weights are directly incorporated into the fitness function), or exploratory regression (when training data is compressed to a smaller subset having a similar information content), or with exploratory weighted regression (when a compressed training subset is weighted with one of

the proposed weighing schemes, and new weights are incorporated into the fitness function).

5.1 Motivation

The goal of this chapter is to assess whether the results of symbolic regression via genetic programming can be improved if the given input-output data set is made balanced. One of the directions for data balancing is modification of the fitness function, which may lead to improvement in the performance of generated solutions. The modification studied in this chapter consists in multiplying the model fitness (or prediction error) in each point by a weight, representing the relative importance of the point among data samples.

As illustrated in section 2 of Chapter 2, data records located in sparse areas of the data space may exert little influence on the learning process. Consequently, trained regression models are conditioned to perform well only on the most typical data records located in well represented areas of the data space. This situation may have a large negative impact on prediction accuracy. In the practice of modeling real 'physical' systems such imbalanced data sets are more a rule then an exception. This fact inspired us to study the possibilities of balancing data by incorporating the "information weight" of a data record into the modeling routine.

The challenge is to substitute the empirical risk functional (2.2),

$$R_{emp}\left(\hat{f}(\mathbf{x})\right) = \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, \hat{f}(x_i)\right),$$

by a weighted empirical risk functional (2.5),

$$R_{emp}\left(\hat{f}(\mathbf{x}), \mathbf{w}\right) = \sum_{i=1}^{N} w_i L\left(y_i, \hat{f}(x_i)\right) / \sum_{i=1}^{N} w_i,$$

in such a way that the performance of individuals obtained by minimization of the modified risk functional improves not only in terms of weighted accuracy, but also in terms of absolute accuracy (unweighed fitness) on new test data.

The problem of improving the effectiveness of regression with adaptive fitness functions (incorporating weights) have been studied extensively by researchers

in the field of supervised learning and evolutionary computation. "Boosting" adaptively assigns weights to data samples based on their difficulty for the modeling, see (Freund, 1995; Freund and Schapire, 1995; Schapire, 1990) for boosting in machine learning and (Drucker, 1997; Iba, 1999; Paris et al., 2001) for boosting in GP. Stepwise adaptation of weights for GP (Eggermont and van Hemert, 2000, 2001) adaptively assigns weights based on the difficulty to meet the constraints imposed on GP solutions. Subset selection methods (Gathercole and Ross, 1994), assign (usually binary) weights to records using various rules, either stochastically or based on performance of GP solutions on data samples. Strange as it may seem, we found no references that assign weights based on the structure of the data, irrespectively of the quality of produced solutions and used modelling technique.

Ideally the weights w_i should not depend on the loss functional $L(y_i, f(\mathbf{x}_i))$ used in (2.2) and (2.5), and on the modelling method, but depend only on the training data $(\mathbf{x}_i, y_i)$, $i = 1 : N$.

The modeling system trained with the modified empirical risk functional will be discouraged to have inaccuracies in the points with higher weights w_i. If thew_i reflect the importance of the corresponding data records and correctly estimate the amount of 'shared responsibility' of records relative to the sparseness of the data space, incorporation of the w_i into the risk functional will force the modeling system to accurately predict under-represented areas of the data space.

Since the weights in the weighted empirical risk functional act as approximations of the data density, their definition has to be related to the mutual distances between data records. Chapter 1 has defined four distance-based weighing procedures, which estimated the relative importance of a data record based on the following attributes:

1. proximity to k nearest-in-the-input-space neighbors (see Eq. (2.8));
2. surrounding by k nearest-in-the-input-space neighbors (see Eq. (2.9));
3. remoteness from k nearest-in-the-input-space neighbors (see Eq. (2.10));
4. local non-linearity relative to k ($k >= d + 1$) nearest-in-the-input-space neighbors (see Eq. (2.12)).

The goal of this chapter is to evaluate the impact of using the proposed

weighing schemes for the purpose of data balancing in the framework of weighted regression.

5.2 Types of weighted regression and GP settings

This section defines different types of weighted regression that should, according to hypotheses of the previous section, outperform standard regression on imbalanced input-output data sets.

Three ways to combine tactics for data weighing and data compression presented in Chapter 2, are suggested:

1. **Weighted regression via GP.** In this general approach, weights based on proximity, surrounding, remoteness, or non-linearity are incorporated into the fitness function directly. Obtaining the weighted fitness functions from the empirical risk functional (2.5) is straightforward. Weighted mean-squared error of a GP individual $\hat{f}$, computed with weights w_i on a set of training points $(\mathbf{x}_i, y_i)$, $i = 1 : N$, is defined as:

$$MSE_{\mathbf{w}}(\hat{f}(\mathbf{x}_i), y_i) = \sum_{i=1}^{N} \left(w_i \left(y_i - \hat{f}(\mathbf{x}_i) \right)^2 \right) / \sum_{i=1}^{n} w_i. \tag{5.1}$$

   Weighted correlation between the output $\hat{f}(\mathbf{x})$, predicted by a GP individual $\hat{f}$, and the observed output $\mathbf{y}$ is defines as:

$$R_{\mathbf{w}}(\hat{f}(\mathbf{x}, y) = \frac{E_{\mathbf{w}} \left(\mathbf{y} \cdot \hat{f}(\mathbf{x}) \right) - E_{\mathbf{w}}(\mathbf{y}) \cdot E_{\mathbf{w}} \left(\hat{f}(\mathbf{x}) \right)}{\sqrt{E_{\mathbf{w}}(\mathbf{y}^2) - E^2_{\mathbf{w}}(\mathbf{y})} \sqrt{E_{\mathbf{w}}(\hat{f}(\mathbf{x})^2) - E^2_{\mathbf{w}}(\hat{f}(\mathbf{x}))}}, \tag{5.2}$$

   where $E_{\mathbf{w}}(\mathbf{y})$ is the weighted average of vector $\mathbf{y}$:

$$E_{\mathbf{w}}(y) = \sum_{i=1}^{n} w_i y_i / \sum_{i=1}^{n} w_i. \tag{5.3}$$

2. **Exploratory regression via GP.** In this approach the training data is first compressed to a balanced subset via the SMITS procedure using proximity, surrounding, remoteness, or non-linearity weights. The selected

Table 5.1: GP Parameters for regression experiments

Number of independent runs	50
Budget per generation	$100N$ function evaluations (N is the number of records)
Total number of generations unless stated differently	250
Population size of REFERENCE runs	100
Archive size	50
Complexity measure	Expressional
Complexity Limit	400 500 for Salustowicz2DBio problem
Population tournament size	7
Archive tournament size	5
Crossover rate	0.95
Mutation rate	0.05
Rate of mutation on terminals	0.3
Range for random real constants	[-10, 10]
Basic Function Set	$+$, $-$, $*$, $/$, $square$ x^{const}, $x + const$, $x \cdot const$, e^x, e^{-x}
Function Set Salustowicz2D	Basic Set, $\sin x$, $\cos x$

subset is then modelled by standard (unweighed) symbolic regression via Pareto GP with **bigger population sizes**. Larger populations enhance exploration of the search space and may therefore improve the effectiveness of evolution, provided that the subset of data captures the information about the underlying response surface.

3. **Weighted exploratory regression via GP.** The third approach is similar to the previous approach in compressing the data to a balanced subset via the SMITS procedure (using proximity, surrounding, remoteness or non-linearity weights). The difference with exploratory regression is in applying **weighted** regression to the selected subset with *bigger population sizes.*

Performance of the indicated types of weighted regression is compared with the performance of the standard symbolic regression on five case studies. The quality of solutions of all experiments is determined by the root mean-squared

error on the test set. For the weighted regression runs *no* weights are used for computing the root mean squared error on the test set. The results of the GP runs are compared with respect to the best root mean squared error on the *test data* of an archive of GP solutions at the last 250th generation.

For each experiment 50 independent GP runs are performed with the same computational budget, where computational budget is defined as a fixed number of function evaluations. The significance of the improvement in the best error is examined via two-sided Wilcoxon-Mann-Whitney tests for comparing the medians of the error samples.

Standard regression is performed by an archive-based Pareto genetic programming system, described in the previous chapter. Experiments related to standard regression are referred to as REFERENCE runs. Settings of the Pareto GP parameters are presented in Table 5.1.

All experiments of this chapter use the same computational budget, measured in the number of function evaluations. The budget is fixed to $100N$ evaluations per generation, where N is the number of records in the data set. For exploratory runs if the data is compressed to a balanced subset of $p\%$ of the original size, the size of the population is set to $\frac{100N}{p} \cdot 100$ individuals.

Comparison experiments are performed on five test regression problems with three diverse response surfaces defined by the following functions:

1. **Salustowicz1d** function (Keijzer, 2003; Salustowicz and Schmidhuber, 1997):
$$f_1(x) = x^3 \exp^{-x} \cos x \sin x (\sin^2 x \cos x - 1); \tag{5.4}$$

2. **Kotanchek** function (Vladislavleva et al., 2008):
$$f_2(x_1, x_2) = \frac{e^{-(x_1-1)^2}}{1.2 + (x_2 - 2.5)^2}; \tag{5.5}$$

3. **Salustowicz2d** function (Vladislavleva et al., 2008):
$$f_3(x_1, x_2) = (x_2 - 5)\hat{f}_1(x_1). \tag{5.6}$$

5.3 Case studies

5.3.1 Salustowicz1d problem

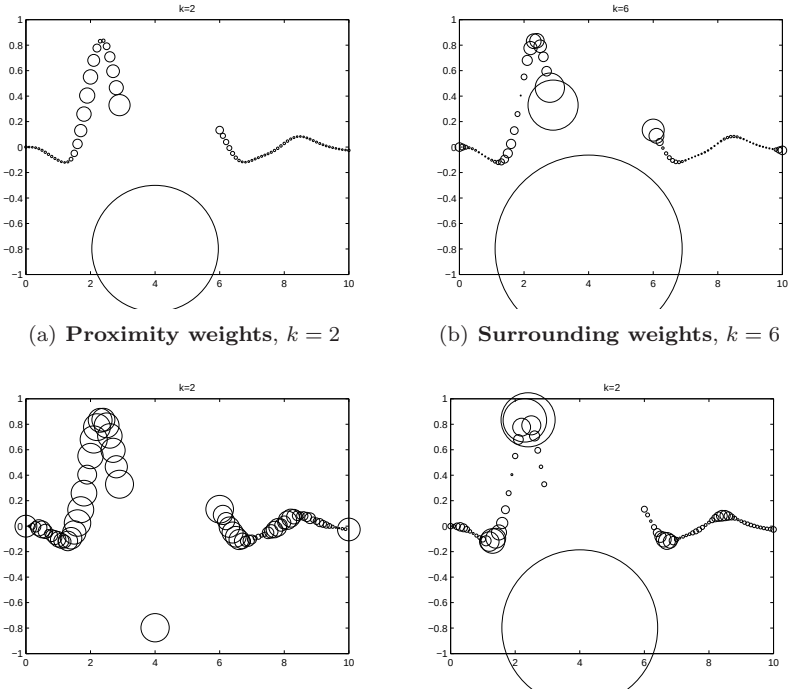
Salustowicz1d function is used for creating one test problem with big gaps in the input space. The input data of the **Salustowicz1d** problem is obtained as a subset of 72 points selected from 100 points sampled uniformly from the interval $[0, 10]$ (see Figures 5.1 and 5.2). For test data we use 1001 values of the **Salustowicz1d** function points sampled uniformly from an interval $[-0.5, 10.5]$.

Salustowicz1d problem with missing data is a good example of the situation where missing an 'outlier' in under-represented region may lead to notoriously erroneous regression models. We illustrate the differences in weighing the data with different weighing functionals in Figure 5.1 and the results of the SMITS-based compression of the **Salustowicz1d** data to 20 points in Figure 5.2. Plot (e) of Figure 5.2 represents the results of compressing the data to a space-filling subset in the input space, obtained by the SMITS procedure with the proximity-X (see Eq.(2.6)) weight with two nearest neighbors. Despite the fact that the outlier point appears in the result of the space-filling compression with proximity-X weight, it may provide insufficient information about areas of the local non-linearity of the Salustowicz1d curve, especially when compared with the results of surrounding- or non-linearity-based compression.

Ten experiments were set up to assess the effects of weighted and exploratory regression on the quality of final solutions when modeling the **Salustowicz1d** problem. The results with respect to the best root mean-squared error of the solutions over 50 independent runs are presented in Figure 5.3.

The REFERENCE runs use the entire data set of 72 points with all weights equal to one. The ProxWEIGHT, SurrWEIGHT, RemotWEIGHT and NonLinWEIGHT stand for the weighted regression via GP performed on the complete data set with the weights obtained via a corresponding weighing procedures. The relative differences of the weights can be observed in Figures 5.1(a) for proximity, 5.1(b) for surrounding,5.1(c) for remoteness, and 5.1(d) for non-linearity weights.

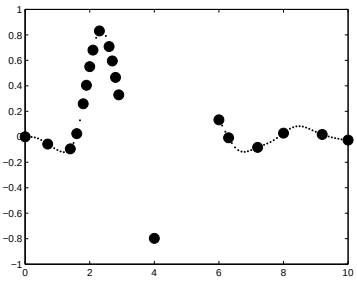
The UniformExplore, ProxExplore, SurrExplore, RemotExplore, and NonLin-Explore in Figure 5.3(b) stand for exploratory regression runs applied to compressed subsets of the **Salustowicz1d** data. These subsets are the results



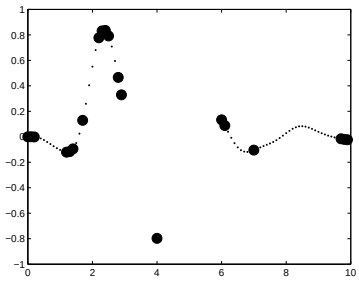
(a) **Proximity weights**, $k = 2$ (b) **Surrounding weights**, $k = 6$

(c) **Remoteness weights**, $k = 2$ (d) **Non-linearity weights**, $k = 2$

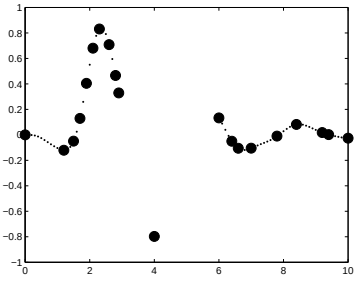
Figure 5.1: Results of Weighing of the Salustowicz1d Data. The figure represents four different weighing schemes applied to a data set of 72 points sampled from the Salustowicz1d function defined by Eq.(5.4). The given data set is an example of a situation, where assigning high weights to the isolated point is necessary for forcing solutions to go through that point and, therefore, to agree with the true underlying response curve in the under-represented areas. Data points are depicted by circles whose radii are proportional to the computed weights. Observe that the proximity, surrounding, and non-linearity weights of the isolated point is considerably higher than weights of other points. Remoteness weights are not suffering from the dominance of that point as much as the proximity and surrounding weights do.



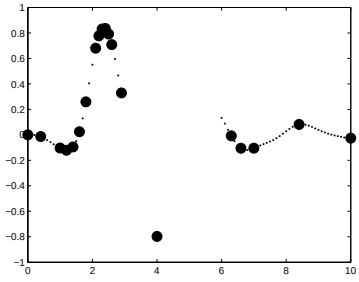
(a) **Proximity Compression,** $k = 2$



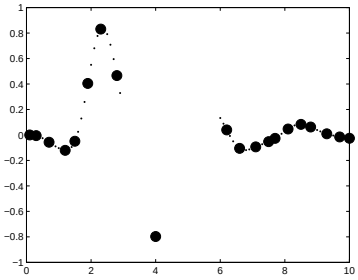
(b) **Surrounding Compression,** $k = 6$



(c) **Remoteness Compression,** $k = 2$

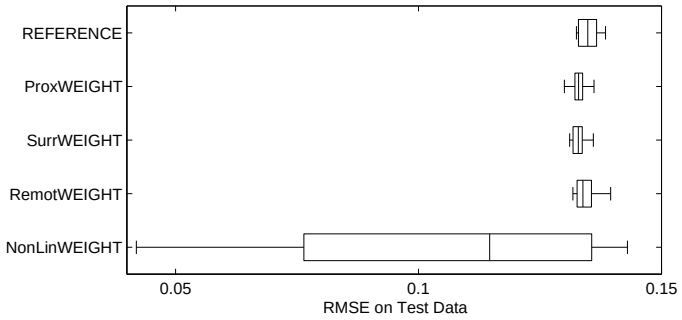


(d) **Non-linearity Compression,** $k = 2$

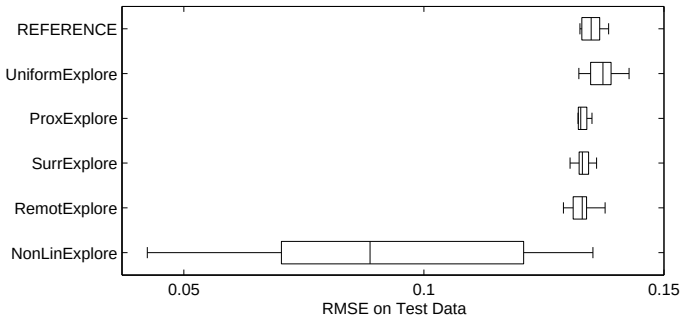


(e) **Proximity-X Compression,** $k = 2$

Figure 5.2: Results of SMITS-based Compression of the Salustowicz1d data. Figure represents the results of compression of the 72 points of the **Salustowicz1d** problem to balanced subsets of 20 points with the SMITS procedure applied to proximity, surrounding, remoteness, non-linearity, and remoteness-X weights. Proximity-X weights used in the last plot lead to a subset of 20 points, which is more or less space-filling in the input space. Note, that the compression to a subset that 'uniformly' covers the input space, may provide insufficient information about the underlying non-linear response surface.



(a) Standard vs. Weighted Regression



(b) Standard vs. Exploratory Regression

Figure 5.3: Fitness of the solutions on the TEST set at the last generation for the Salustowicz1d problem. Plot 5.3(a) represents a comparison of the reference GP runs with the runs minimizing a weighted fitness (WNMSE) for four weighing schemes: proximity, surrounding, remoteness and non-linearity. Plot 5.3(b) represents the comparison of the reference runs with the exploratory runs based on the SMITS compression of the data via the four above-mentioned weighing schemes. For all exploratory runs the original data set of 72 points was compressed to 20 points (the training data are plotted in Figure 5.2(a), 5.2(b), 5.2(c), 5.2(d), 5.2(e) for ProxExplore, SurrExplore, RemoteExplore, NonLinExplore, and UniformExplore runs respectively). The population size of the reference runs is 100 equations, the population size of the exploratory runs is 360 equations, the total budget per generation is 7200 function evaluations for all runs. The boxplots are built based on the results of 50 independent runs performed for each experiment.

of the SMITS-based compression to 20 points with one of the five weighing functionals. We plot them in Figure 5.2(a) for proximity, 5.2(b) for surrounding, 5.2(c) for remoteness, 5.2(d) for non-linearity weights, and 5.2(e) for proximity-X weights. Only those subsets are used as the training data for the exploratory runs. The UniformExplore denotes the exploratory regression applied to the subset of 20 points obtained via the SMITS procedure using the proximity-X weight and two nearest neighbors, that cover the input interval $[0, 10]$ sufficiently uniformly.

Since the size of the training set in Exploratory runs is 3.6 times smaller than the size of the original data used for training the REFERENCE runs, the population size of the exploratory runs is increased to 360 individuals from the reference size of 100 individuals.

The pair-wise Wilcoxon-Mann-Whitney rank-sum tests show that the runs for weighted regression corresponding to ProxWEIGHT, SurrWEIGHT and NonLinWEIGHT produce median root mean squared error on the test data, which is significantly smaller than the one of the REFERENCE runs. The p-values for rejecting the null-hypothesis of equal medians (for the pair-wise comparison with the reference) are 0.0006, 0.0009, and 0.0002 for the proximity, surrounding, and non-linearity weights respectively. The non-linearity based weighted regression also significantly outperforms regression based on proximity, surrounding and remoteness weights (p-values for pair-wise Wilcoxon rank-sum tests are 0.0433, 0.0398, and 0.0012 respectively).

The exploratory regression UniformExplore on the subset, uniformly covering the input space (plotted in Figure 5.2(e)) is significantly inferior to standard regression of the REFERENCE runs with respect to the best error on the test set (p-value for the null hypothesis is 0.0042). All other exploratory runs, ProxExplore, SurrExplore, RemExplore and NonLinExplore, show significant superiority over the REFERENCE runs with p-values 0.0003, 0.0051, 0.0125, 0.0000 respectively. All exploratory runs are statistically inferior to NonLinExplore runs with zero p-values.

The results demonstrate that the maximum improvement of the weighted regression over the standard regression on **Salustowicz1d** problem is obtained with the non-linearity weights. Exploratory regression via GP significantly improves the standard regression for all weights, but the improvement of non-linearity- based exploratory regression is maximal. The latter also statistically

outperforms the weighted regression on non-linearity weights with a small p-value, namely 0.0412.

5.3.2 Kotanchek100 problem

This and the two following case studies are created using the Kotanchek function with various input sets. **Kotanchek100** problem contains a set of 100 points with inputs sampled *randomly and uniformly* from the interval $[0.2, 3.8] \times [0.2, 3.8]$ and outputs computed with the Kotanchek function, see Figure 5.4(a).

The test set for **Kotanchek100** problem consists of 4225 points and forms a uniform mesh on the input interval $[0, 4] \times [0, 4]$.

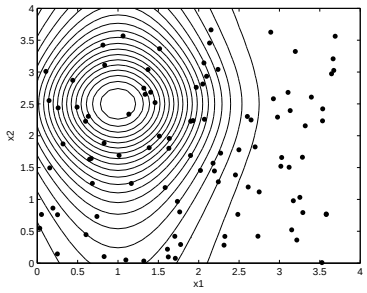
The purpose of experiments on the **Kotanchek100** problem is to test whether weighted regression and exploratory regression based on the non-linearity weights still outperform the standard regression in the case of a standard random uniform sampling of a two dimensional input space.

Figure 5.4(c) demonstrates the best root mean squared error of the final GP solutions over 50 independent runs. ProxWEIGHT and SurrWEIGHT stand for weighted regression with proximity and surrounding weights for three nearest-in-the-input-space neighbors. NonLinWEIGHT stands for weighted regression with non-linearity weights for the same neighborhood size, see Figure 5.4(b). NonLinExplore stand for exploratory regression applied to a subset of 30 points obtained with the SMITS-based compression of 100 **Kotanchek100** points. The population size of the exploratory runs is 332 individuals. The computational budget is equal to 10000 function evaluations per generation for all experiments.

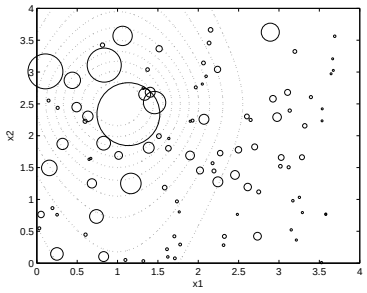
The Wilcoxon rank sum tests did not reveal statistical differences between the REFERENCE runs and the runs for weighted regression on proximity and surrounding weights. The solutions of NonLinWEIGHT and NonLinExplore runs are statistically better on the test set than the reference solutions with p-values equal to 0.0176 and 0.0010. There is no evidence that the solutions of NonLinWEIGHT and NonLinExplore are statistically different.

5.3.3 KotanchekImbalanced problem

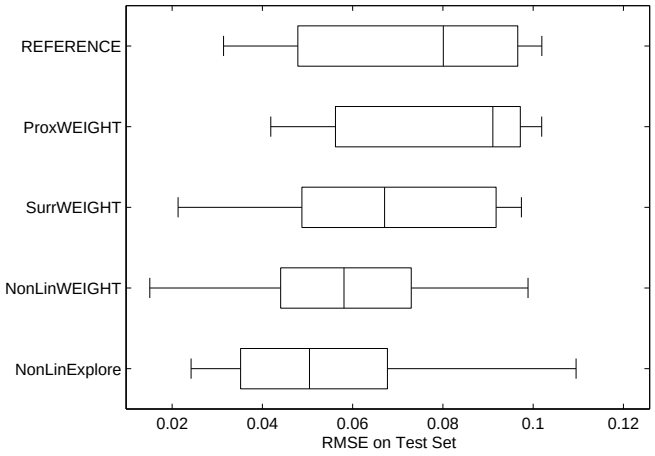
Training data for the **KotanchekImbalanced** problem consist of 100 points with inputs sampled *non-uniformly* from the interval $[0.2, 3.8] \times [0.2, 3.8]$ in such a way, that only one half of the records captures the non-linearity of the response



(a) **Kotanchek100 Data, 100 points**



(b) **Non-linearity weights,** $k = 3$



(c) **Results on the Test Data**

Figure 5.4: Weighted and Exploratory regression on the Kotanchek100 problem.

surface of the Kotanchek function. The test set for **KotanchekImbalanced** problem is the same as for the **Kotanchek100** problem and consists of 4225 points and forms a uniform mesh on the input interval $[0, 4] \times [0, 4]$.

The empirical results obtained for two previous case studies have indicated that weighing and SMITS-based compression obtained with the non-linearity weight can significantly improve the quality of GP solutions for small and reasonably sampled data sets, where points are uniformly surrounded by k nearest-in-the-input space neighbors. The **KotanchekImbalanced** is an example of a very imbalanced data set, such that the distances to the three nearest-in-the-input-space neighbors vary significantly over the data samples. Due to this fact, weighing based on the non-linearity may not be beneficial for regression.

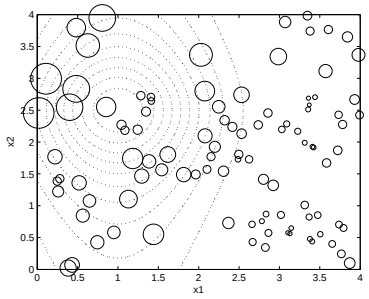
We test this hypothesis in Figure 5.5(c). The box-plot confirms that the weighted regression based on the non-linearity weights does not produce a statistically significant improvement of the solutions on the test set compared with standard regression on the KotanchekImbalanced problem. On the contrary, the REFERENCE runs are statistically better that the NonLinWEIGHT runs with p-value of 0.0091.

The ProxWEIGHT and SurrWEIGHT runs stand for the weighted regression with proximity and surrounding weights with three nearest-in-the input space neighbors. Despite the imbalancedness of the data set, the ProxWEIGHT and SurrWEIGHT runs do show the significant improvement over the REFERENCE runs with p-values equal to 0.0051 and 0.0000 respectively.

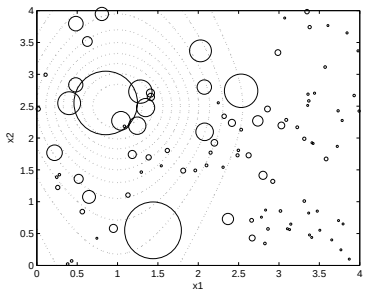
The exploratory regression runs are performed on a subset of 50 points, obtained with the SMITS-based compression of **KotanchekImbalanced** data using proximity weights for ProxExplore, and non-linearity weights for NonLin-Explore. The population size of exploratory runs is 200, the computational budget is 10000 function evaluations per generation. Both exploratory regression experiments based on proximity and non-linearity weights show significant improvement over the reference runs with zero p-values.

5.3.4 KotanchekBio problem

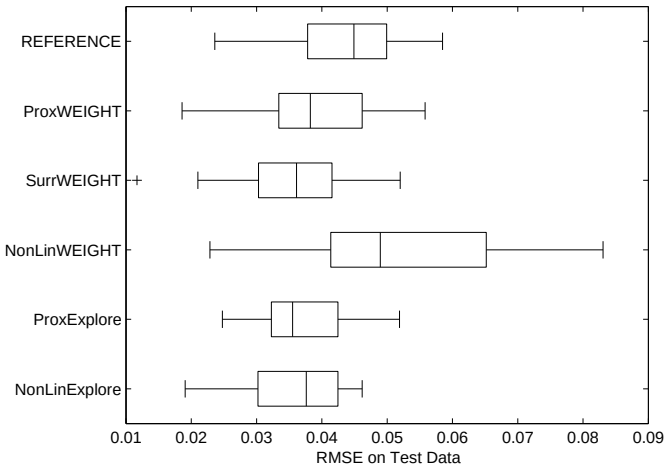
The training set for **KotanchekBio** problem consist of 3931 records with inputs corresponding to an industrial set of real measurements scaled to the interval



(a) **Proximity Weights,** $k = 3$



(b) **Non-linearity Weights,** $k = 3$



(c) **Results on the Test Data**

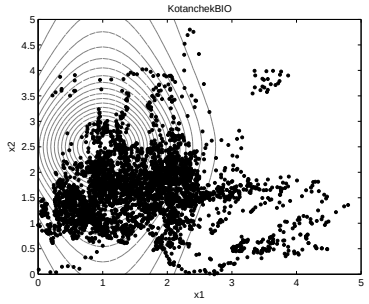
Figure 5.5: Weighted and Exploratory regression on the KotanchekImbalanced Problem.

$[0, 4.8] \times [0, 4.8]$. The input set comes from the study on the Biox water purification plant and the relation of plants' feeding systems to the amount of the floating sludge. Two Biox-related input variables and the corresponding undesigned set of physical measurements were selected as an example of real-life imbalanced input data. The responses for given 3931 records is computed using the Kotanchek function (Eq. 5.5).

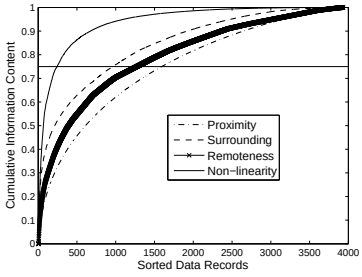
KotanchekBio problem is heavily over-sampled, see Figure 5.6(a). The expectation for such over-sampled data is that exploratory regression should produce significantly better solutions if the compressed training subset captures the information content of the data.

The weighted, exploratory, and weighted exploratory symbolic regression of the KotanchekBio data is performed using the non-linearity weights computed with five nearest-in-the-input-space neighbors (in some areas of the data space, data sampling is so dense or contains duplicate points that the least-square planes approximating a smaller number of nearest-in-the-input-space neighbors (4 and 3) are ill defined). For the training data of exploratory runs NonLinExplore and NonLinExploreWEIGHT we compressed the **KotanchekBio** set to a subset of 250 points with the SMITS procedure using non-linearity weights, see Figure 5.6(c). The cumulative information content of the obtained subset is 0.75 (see Figure 5.6(b)). The population size used for modeling the compressed subset is 1572 individuals (for both NonLinExplore and NonLinExploreWEIGHT experiments). NonLinExploreWEIGHT experiments use the selected subset of 250 points weighted with non-linearity weights for three nearest-in-the-input-space neighbors. The weights of the compressed subset are illustrated in Figure 5.5(b).

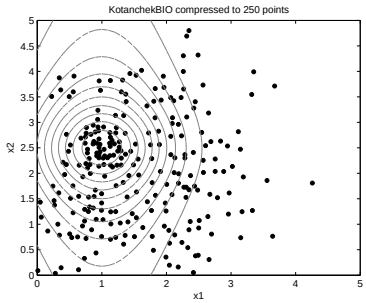
Analysis of empirical results on the **KotanchekBio** problem leads to surprising conclusions. Not only the exploratory runs, NonLinExplore and NonLinExplore- WEIGHT significantly outperform the REFERENCE runs with zero p-values, but also do all weighted regression runs ProxWEIGHT, SurrWEIGHT, NonLinWEIGHT, with p-values of 0.0001, 0.0002, and 0.0000 respectively. The mean and the median of the best root mean-squared errors of the NonLinExploreWEIGHT runs are smaller than those of NonLinExplore runs, but two sided ANOVA tests and Wilcoxon-Mann-Whitney tests do not reveal statistically significant differences between these two experiments. Both exploratory runs outperform the weighted regression runs with zero p-values.



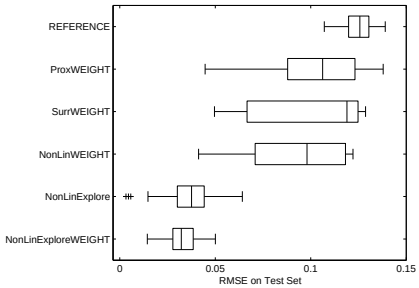
(a) **KotanchekBio Data, 3931 points**



(b) **Cumulative Information Content,** $k = 5$

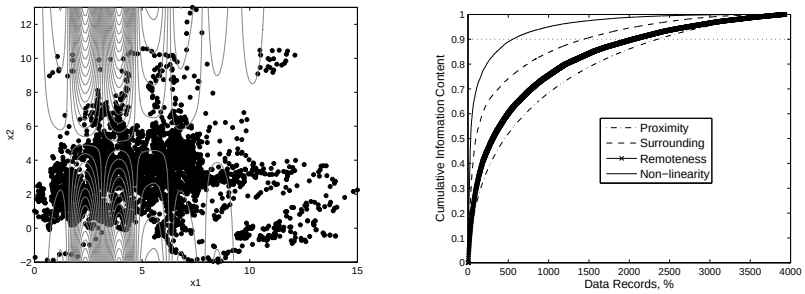


(c) **Non-linearity Compression, 250 points**



(d) **GP results**

Figure 5.6: Weighted, Exploratory and Exploratory Weighted regression on the KotanchekBio Data using the non-linearity weights. The quality of GP solutions can be significantly improved at the same computational budget if non-linearity weights are incorporated into the fitness function (NonLinWEIGHT). Such weighted regression can be further improved by compressing the KotanchekBio data with the SMITS procedure to 15.72% of the original size and applying to it exploratory regression with bigger populations (NonLinExplore). We hypothesize, that further weighing of the compressed subset may again cause the improvement in the quality of final solutions (NonLinExploreWEIGHT). However, for this data set no statistical difference is observed between NonLinExplore and NonLinExploreWEIGHT runs. We explain this by the fact that the errors obtained by the NonLinExplore runs are already very low, and the GP system has difficulties in outperforming such good solutions with statistical significance.



(a) **Salustowicz2dBio Data** (b) **Cumulative Information Content**

Figure 5.7: Salustowicz2dBio problem with 3931 records and the Cumulative Information Content of rankings obtained with the smits procedure and different weight functionals and five nearest-in-the-input-space-neighbors.

5.3.5 Salustowicz2dBio Data

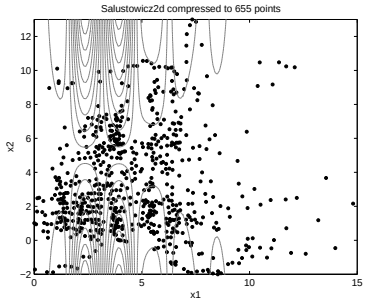
For the set of training inputs of the **Salustowicz2dBio** problem the same real-life Biox set was used as for the **KotanchekBio** problem, but now scaled to the interval $[0, 15] \times [-2, 13]$. The training response is computed by the Salustowicz2d function given by Eq.(5.6). Input data and the contour plot of the Salustowicz2d function are presented in Figure 5.7(a).

The test data for the **Salustowicz2dBio** problem contains 5000 records, and forms a uniform mesh on the interval $[0, 15] \times [-2, 13]$.

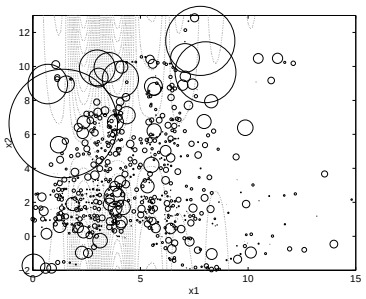
Salustowicz2dBio data is a very difficult regression problem, since most of the 3931 data points are sampled from a relatively 'flat' region of the response surface, see Figure 5.7(a). To prevent convergence to a trivial solution we model this problem with a correlation as fitness function and add sine and cosine to the set of basic functions; see Table 5.1. Short runs of 160 generations are performed, with a fixed budget equal to 3931×100 function evaluations per generation.

For exploratory runs NonLinExplore the **Salustowicz2dBio** data were ranked with the SMITS procedure using non-linearity weights with five nearest-in-the-input-space neighbors. From the ranked set a subset of the top 655 points was selected with a cumulative information content of 0.9, see Figures 5.8(a) and 5.7(b). This compressed subset was used as the training set for exploratory experiments runs with populations of $\lfloor 3931/655 \cdot 100 \rfloor = 600$ individuals.

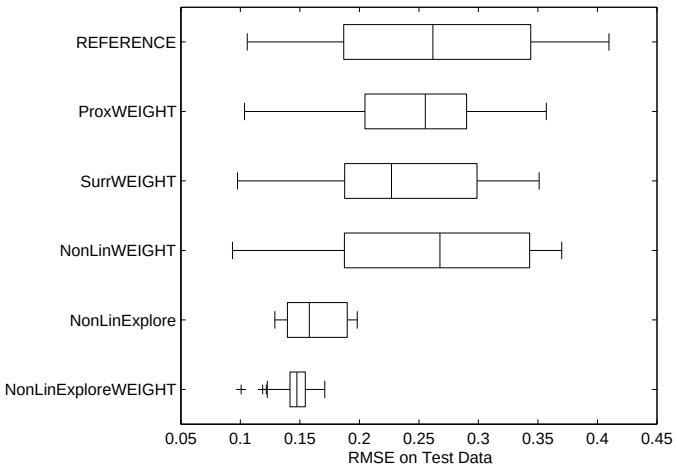
For weighted exploratory regression (NonLinExploreWEIGHT runs) we use



(a) **Salustowicz2dBio, 655 points**



(b) **Non-linearity weights of 655 points**



(c) **GP Results**

Figure 5.8: Weighted, Exploratory and Exploratory weighted regression on the Salustowicz2dBio data. Figures (a) and (b) present the results of the SMITS-based compression of the Salustowicz2dBio data to subset of 655 and and corresponding non-linearity weights of the compressed set. Figure (c) presents the box-whisker plots of the GP results. The subset of 655 points is used as the training set for NonLinExplore experiments with populations of 600 individuals. The NonLinExploreWEIGHT experiments are trained on the same subset of 655 points but with the non-linearity weights depicted in plot (b).

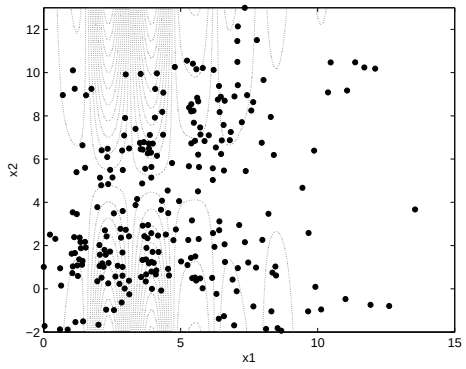


Figure 5.9: Compression of the Salustowicz2dBio data to 250 points with the smits procedure using non-linearity weights and five nearest-in-the-input-space neighbors.

the compressed set of 655 points together with computed non-linearity weights with three nearest-in-the-input-space neighbors; see Figure 5.8(b). The weighted exploratory regression is performed with populations of 600 individuals.

The results of the SMITS-based compression of **Salustowicz2dBio** data with non-linearity weights to 250 points are presented in Figure 5.9. The resulting subset can be compared with the compression of **KotanchekBio** data to 250 points with non-linearity weights and the same number of nearest-in-the-input-space neighbors, plotted in Figure 5.6(c). Comparing the compressed subsets, we can visually confirm that the SMITS-based compression using the non-linearity weights does better sample the areas of the response surface that have local non-linearities. Comparing Figures 5.9 and 5.6(c) we can observe the big difference in the compressed subsets of 250 records, corresponding to two data sets, which are originally identical in the input space (after scaling), but have response surfaces of different non-linearity.

The performance of regression experiments is summarized in Figure 5.8(c). The results indicate that the weighted regression on the full **Salustowicz2dBio** data does not cause improvements of final solutions. The exploratory regression experiments NonLinExplore and NonLinExploreWEIGHT are superior to the REFERENCE and NonLinWEIGHT runs significantly with virtually zero p-values. Weighted exploratory regression (NonLinExploreWEIGHT) is statistically superior to exploratory regression experiments (NonLinExplore) this time,

although with a relatively high p-value of 0.0302.

5.4 Summary and guidelines

This chapter has presented the possibilities of improving symbolic regression via genetic programming by changing the model evaluation routine through incorporation of the information about the relative importance of the input-output data records into the definition of the error function.

Three methods of changing the fitness function are presented. All of them exploit the procedures for data weighing and data balancing introduced in Chapter 2.

Based on the results of applying the proposed methods to five regression problems, the following preliminary conclusions can be drawn:

1) Weighted regression consisting of direct incorporation of weights into the fitness function, can significantly outperform standard regression. The best improvements are observed when using the non-linearity weights for the case of relatively balanced data sets, and when using surrounding weights for the case of very imbalanced data sets.

2) The iterative SMITS procedure (ranking the input-output data set in the order of decreasing importance) can be successfully used for compressing the input-output data to smaller subsets with a similar information content. This feature may considerably enhance the analysis and modeling of large imbalanced data sets with highly non-linear response surfaces, when more modeling effort is applied to smaller subsets of data. This statement is validated by statistical superiority of exploratory regression experiments over the standard regression experiments on all case studies.

3) If input-output data is heavily over-sampled, it can be beneficial to compress it with the SMITS procedure using non-linearity, or surrounding weight to a smaller balanced subset of a similar information content and then apply weighted regression to the compressed subset, using non-linearity or surrounding weights.

4) For data with d-dimensional input space, $k = d + 1$ appears to be an appropriate default setting for the number of nearest-in-the-input-space neighbors for the general-purpose weighing routine, especially with the non-linearity weight. If surrounding or remoteness weights are used, reducing the

neighborhood size to one neighbor may be beneficial for improving efficiency of weight calculations. The weights need to be computed only once, in a pre-processing stage before the modeling stage. However, in dynamic environments re-balancing will be required when the data set changes. In such situations, reducing the neighborhood size to one neighbor (and using surrounding weights) will be crucial for efficiency.

Figure 5.10 summarizes the guidelines for data balancing at the current state of research (presented in this chapter and in Chapter 1). When the data set is given, it is usually easy to say whether it is clearly under-sampled or not. If it is under-sampled and nothing else is known about the underlying function or areas of the design space that are of any special importance - no balancing should be performed before applying standard regression (all weights in the fitness functions are equal to one).

If the data is not heavily under-sampled, and compression is not required, weighted regression can produce results significantly better than those of standard regression. Some intuition and more research are required to decide on the weighing scheme, but if the data is not too imbalanced, then it is worthwhile to use the non-linearity weights. Otherwise, surrounding, remoteness, or proximity weights can be used.

If computational resources allow, it is better to compute proximity, surrounding, remoteness, and non-linearity weights all together. Their sorted profiles provide insight into the structure of the data.

For over-sampled data sets we recommend using the SMITS procedure on one of the four weighing schemes for compressing the data to a smaller balanced subset. This subset can be either used in the standard regression, or it can be weighted again. Since the compressed balanced subset will not be imbalanced by definition, we advise using non-linearity weights for balancing the subset points. In high-dimensional data, however, especially coming from measurements, the surrounding weight seems to be a better choice than the non-linearity weight, whose meaning deteriorates if nearest-in-the-input-space neighbors are not surrounding the points, where the weights is computed uniformly.

As summarized in Chapter 2, for data sets of high dimensionality the use of fractional distance metrics (Aggarwal et al., 2001; Doherty et al., 2004; François

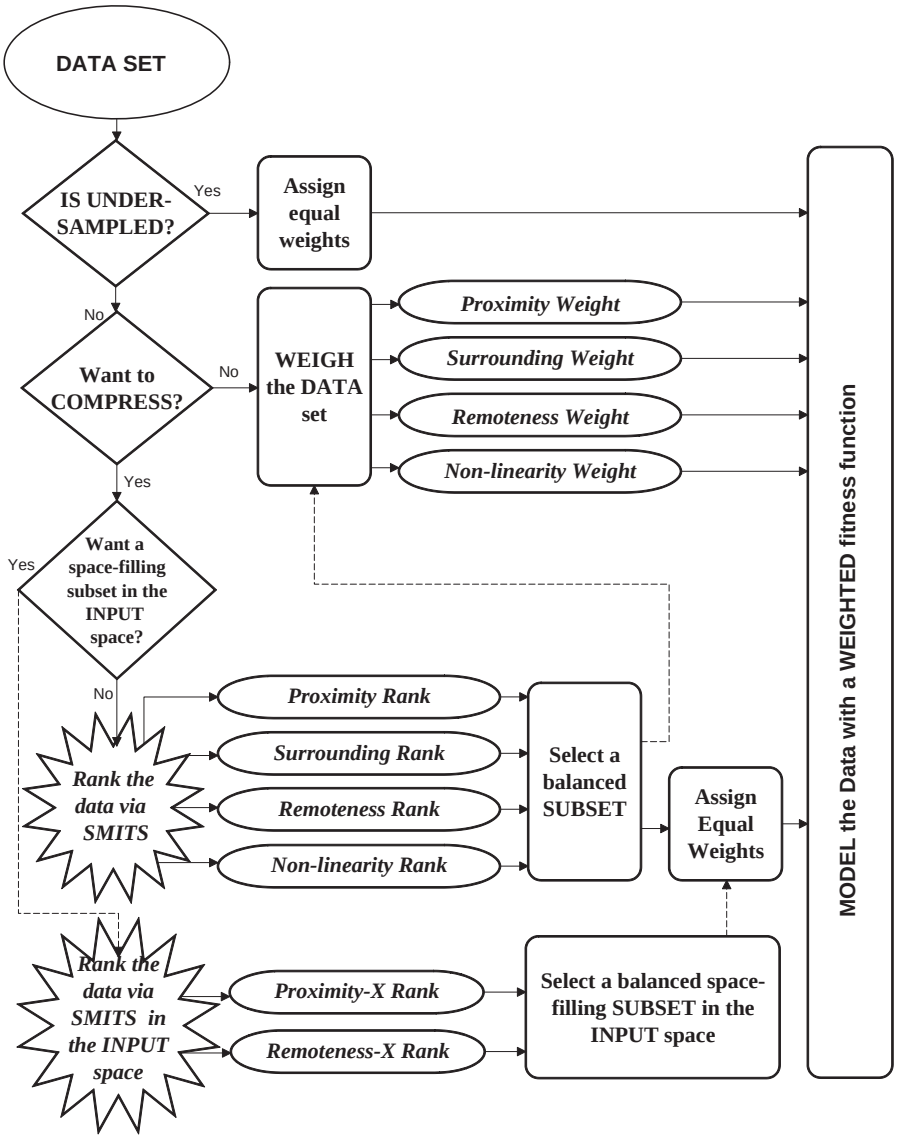


Figure 5.10: Guidelines for data balancing relative to the user request.

et al., 2007; Jin et al., 2003) is strongly advised for finding nearest or nearest-in-the input space neighbors. We also recommend to perform data balancing in the subspace of significant variables[1], and indicate that further research is needed to study the effects of including spurious variables into the weighing procedures.

In general the case studies demonstrated that data balancing can improve the results of symbolic regression in one or another way. This is a motivating incentive to continue research on data balancing and to keep looking for new opportunities to further enhance the capabilities of symbolic regression via genetic programming.

[1]When high-dimensional data needs to be weighted, short screening GP runs may be needed for initial variable sensitivity analysis and dimensionality reduction. In general, we suggest to incorporate automatic re-balancing routines into the automatic on-line sensitivity analysis.

6

Model Evaluation through Goal Softening

This chapter presents a new enhancement of the old idea of evaluating regression models on subsets of training data to speed-up evolution, to improve robustness, and to escape from local optima of the fitness landscape. The new view on partial fitness evaluations consists in the use of the principles of ordinal optimization. The latter, when applied to iterative search in vast search spaces, suggests to start evaluating large quantities of potential solutions coarsely, and gradually increase the fidelity of evaluations, once potentially suitable areas of the search space are identified.

The chapter illustrates that focusing on ranking in combination with goal softening is a very powerful way to improve the efficiency and effectiveness of archive-based evolutionary search. The presented strategy consists of partial fitness evaluations of individuals on random subsets of the original data set, with a gradual increase in the subset size in consecutive iterations. A series of experiments performed on three test problems indicates that those evolutions that started from the smallest subset sizes of only 10% of the original data size consistently provide results that are superior in terms of the goodness of fit, consistency between independent runs, and computational effort. The results also suggest that solutions obtained using the new approach are less complex than the solutions of the standard regression.

6.1 Motivation

Identification of a mathematical expression describing the behavior of an unknown system is a challenging task. The search space of a real-life symbolic regression problem is tremendously huge. The lack of any significant structure makes the navigation through this search space intrinsically hard. Noise, multi-dimensionality of data, complexity of underlying relationships also contribute to making the search inherently difficult.

In the evolutionary framework, three (over-lapping) ways can be distinguished to reduce the time needed to find an appropriate solution or a solution ensemble in a large search space.

The first approach is to transform the search space to a smaller one or a smoother one. Examples of such transformations are changing the representation of individuals, parsimony pressure, introducing multi-objective optimization, elite-based selection, automatical selection of driving variables, and adaptive fitness function.

The second approach is to improve the navigation through the search space to increase the probability of discovering promising areas of solutions with sufficient quality. Examples of this approach are the optimization of genetic operators, again changing the representation, alternating complexity measures in the multi-objective optimization framework, etc.

The third approach is to speed-up the routines for fitness evaluation. It has been studied extensively by researchers practicing evolutionary computation since the early 90s. Here three main directions to reduce the computational effort can be distinguished[1]:

1. evaluating all individuals on *subsets* of training data of a certain size;

2. *racing* individuals on subsets of training data to focus computational effort on guaranteed winners of selection;

3. *backward chaining* an evolutionary algorithm for evaluating only those individuals that are sampled for the tournaments.

Zhang and Veenker (1991) presented an inspiring idea on efficient training of neural networks on reduced training sets. Gathercole and Ross (1994) enhanced

[1]Besides these enhancements the question of an actual speed-up of fitness evaluation by using registers or graphic processing units has been studied extensively in the last years; see Brameier et al. (1999) and Harding and Banzhaf (2007).

classification via genetic programming with three *subset selection* schemes for training data - dynamic, historical, and random subset selection. This approach to boosting classification was further developed by Curry and Heywood (2004), who presented hierarchical dynamic subset selection for classifying large data sets (see also (Song et al., 2005)), and by Curry et al. (2007), who enhanced the methodology by hierarchical balanced-block dynamic subset selection, again for classification.

Teller and Andre (1997) came up with a rational allocation of trials - a general framework for automatic subset selection, which for the first time applied *racing* to regression models generated through evolutionary computation and allowed big savings in the computation time.

Zhang and Cho (1998) accelerated genetic programming with active subset selection for evolution of multi-agent strategies. The active subset selection consisted in incremental data inheritance, where data subsets are nested and are evolved through data cross-over in parallel with the evolution of individuals, see also (Zhang and Joung, 1999). Zhang (1999) introduced Bayesian genetic programming as a framework for evolving "compact programs fast without loss of their generalization accuracy" that embraced the incremental data inheritance.

Since 1997 co-evolutionary techniques for enhanced problem solving have received much attention (e.g., (Ahluwalia and Bull, 2001; Bucci et al., 2004; Ficici and Pollack, 2001; Rosin and Belew, 1997)). Papers by (Bongard and Lipson, 2005; Lemczyk and Heywood, 2007; Schmidt and Lipson, 2006) report large reduction in computation time by using a co-evolutionary framework, where combinations of training records of a fixed (small) size are co-evolving together with regression models.

Lasarczyk et al. (2004) introduced dynamic subset selection based on a fitness case topology; this is the first approach to selecting points that provide the most diverse information about the problem structure based on the performance of population individuals.

Poli (2005) suggested backward chaining for evolutionary algorithms with tournament selection to save computation effort of individuals that are never selected for tournaments, see also (Poli and Langdon, 2005) for empirical study of backward chaining in genetic programming. The biggest savings of computation time can be achieved when very low selection pressure is used with big population sizes in short runs.

Some of the proposed schemes are very generic and can be used simultaneously without considerable change in evolutionary strategy to achieve a meta-speed up in function evaluations. For example, rational allocation of trials of Teller and Andre (1997), topology based selection of Lasarczyk et al. (2004), and backward chaining GP of Poli and Langdon (2005) are smart approaches for avoiding unnecessary calculations. Using one or another technique does not forbid application of sub-sampling schemes, aimed at evaluating individuals on a (common) subset of the training set over several generations in a row.

We propose an approach based on the ideas of the theory of Ordinal Optimization (for the best description see (Ho, 2000)). The ideas of ordinal optimization, and more specifically of goal softening and ranking, not only fully concur with common sense and intuition about evolutionary search, but may couch them in the strong language of mathematics.

A first simple application of ordinal optimization to genetic programming is presented in Smits and Vladislavleva (2006). This chapter further extends and analyzes the concept of ordinal Pareto GP presented in (Smits and Vladislavleva, 2006). It also applies additional goal softening to archive-based Pareto genetic programming and illustrates that further reduction of the computational budget is possible while effectively keeping the same quality of the results. We present a preliminary examination of the impact of allocating more computational budget to the exploration stage of a GP run without changing the total budget for the run.

6.2 Goal Softening and Ranking

The concept of Ordinal Optimization is extensively described in (Ho, 2000). This section presents a simplictic summary of the main ideas; see also (Ho, 2000; Ho et al., 2000; Lau and Ho, 1997). The theory rests on two basic tenets:

- It is easier to find a good enough solution with high confidence than the best solution for sure. Such *goal softening* helps to smooth and direct the search.

- It is easier to determine Order than Value, or, in other words, it is easier to determine whether $A > B$ than to determine A and B exactly.

When solving *hard* problems by computationally expensive (e.g. evolutionary) search-based methods, we argue that quickly narrowing down the search for an optimum to a 'good enough' subset of the search space is more important than accurate estimation of performance of potential solutions during the search.

Goal Softening

In real-life problems the true optimum is often unattainable, and the compromise is 'good enough' solutions. This substitution of the best solution by a good enough subset of solutions, or of being sure by being confident with high probability is an example of softening the optimization goals.

Order vs. Value

Imagine we have to quickly select the fastest runner of two. Instead of monitoring them separately for weeks and measuring the speed on various distances, we will let them compete with each other, and will choose the one who arrives at the finish first. In general, it is much easier to determine whether $A > B$, i.e. to determine the order, than to exactly estimate the difference between A and B under multiple conditions, i.e. to examine their cardinal values. Thus, although there is less certainty associated with the decision based on the ordinal approach, it is obtained at a much lower cost compared with the approach based on exhaustive evaluations.

6.3 Goal softening and ranking in GP

6.3.1 Ordinal ParetoGP: better solutions with more effort

Motivation

Considering the vastness of the search space, we can improve the effectiveness of our evolutionary search procedure in either of two ways. One way is to try to get solutions of a similar quality at a lower computational cost. A second way is to try to get solutions of a better quality at the same or similar computational cost. We question whether we need exhaustive fitness evaluations to evolve solutions of a similar quality. If fitness evaluation can be 'softened', to what extent can we then decrease the size of the subsets of the data to perform this evaluation?

We also wonder, what can be gained in terms of system performance and the computational budget, by evaluating more potential solutions at a lower cost and by gradually improving the fidelity of the fitness evaluations in the course of an evolution.

In symbolic regression via GP the bulk of the computational effort is spent on fitness evaluation of the individuals[2]. In many cases all available records of a given data set are used to determine the fitness of every individual. These fitness values are then used to rank all models, and then decide who gets the right to propagate and who does not. As emphasized in (Smits and Vladislavleva, 2006), it is not really critical what sort of selection system is being used - the essence is that all that is required is an ordering of the equations in terms of their performance (actually what we really need is not an ordering in terms of their current performance but an ordering in terms of their potential to generate better offspring).

Initial empirical studies of (Smits and Vladislavleva, 2006) confirmed the general trend of awareness that the effectiveness and the reproducibility of the search can be improved considerably by introducing incomplete fitness evaluations. The winning strategy used partial fitness evaluations on random subsets of the original data set, while gradually increasing the subset size and decreasing the population size in consecutive generations. Partial fitness evaluations with stochastic sampling started with the subset size of 10% of the original size, and increased it to 100% during the first 200 of the total of 250 generations. At the same time the population size was decreased linearly from 1000 to 100 over the first 200 generations. The archive size was kept constant at 100 models at all times. These ordinal ParetoGP runs of 250 generations outperformed standard ParetoGP runs of a 1000 generations, both in terms of the final fitness and the consistency of 30 independent replicates.

The next section presents an explanation of the improved performance observed in (Smits and Vladislavleva, 2006) by examining the influence of partial fitness evaluations on the ranking of individuals and observing the resulting change in the balance of exploration versus exploitation.

[2]This holds for data sets of medium and large size, which are our main interest.

Better solutions because of better exploration

Instead of evaluating every model exhaustively using the full data set, the focus is moved to the use of a subset of the available data to rank the models. A random subset used to estimate the fitness of an individual introduces a certain level of noise in this estimate. By repeatedly selecting different subsets of a given size to determine the fitness of a given individual and examining resulting distributions, we can get an idea of the level of uncertainty. These distributions obviously depend on the size of the subset (smaller subsets will cause distributions with bigger variance), but also depend on the individual itself (fitter individuals will have smaller distributions), see Figure 6.1.

In Figure 6.2 we present the normal approximations of the histograms of the distributions, presented in Figure 6.1. The means are equal to the true fitness of the individuals, and the standard deviations are inversely proportional to the subset size and to the quality of the model[3].

When the true fitness distributions of two individuals overlap (see Figure 6.2), and we have to rank individuals based on fitness evaluations on one subset, we can obviously make an error in taking the decision which is the best of the two. These 'mistakes' in the ranking are the first reason for enhanced exploration. A given individual can outrank another individual in the population but in addition that individual could also outperform and replace other individuals that are already in the archive. Since individuals that would be of lower fitness can now end up in the archive and contribute to creating offspring in the next generation, this increases the level of exploration in the search.

Another reason for better exploration is the fact that the use of subsets allows us to evaluate more models with the same computational budget. If we use subsets of a given size, sampled uniformly at random, they will be different for every generation. These will act as successive screens that every model will need to pass in order to survive in the long term. The result is a regularizing effect and increased robustness of the final models because there is less opportunity for pathologies or for learning the noise that may be present in the data.

[3]This is true for smaller subset sizes (10-75%). For bigger subset sizes the distribution of model fitness deviates from the normal, and can be modeled as a Weibull distribution.

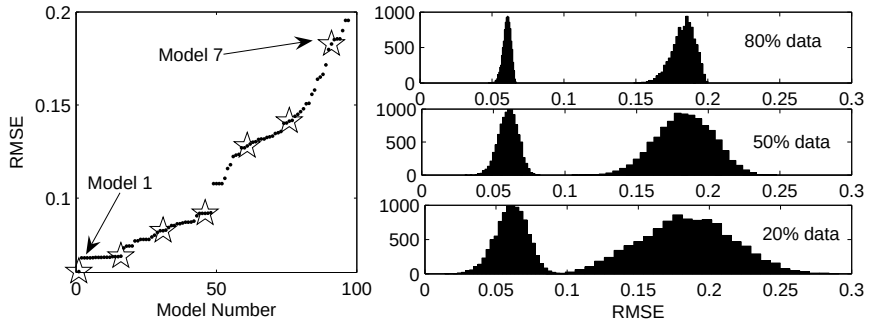


Figure 6.1: Partial fitness evaluations of archive models. The left plot illustrates the sorted fitness values of an archive of a Kotanchek run at the last generation. Seven models (emphasized as stars) are selected from 100 archive individuals for further analysis. The plot at the right size shows the histograms of fitness distributions of model 1 and model 7 (the best and the worst from the selected set of models), computed on 10000 random subsets of 20, 50, and 80% of the training data. We see that the widths of the histograms, i.e. the deviation of the estimated fitness from the true fitness, are inversely proportional to the size of the subset, and to the quality of the individual itself. E.g., model 1, which is the best one in the archive, has the narrowest fitness distribution, which becomes narrower as the subset size increases.

More effort because of archive re-evaluations

In the new approach the number of function evaluations is kept constant per generation. Therefore, by design, the computational budget spent on fitness evaluations of every new population does not change. An additional budget, however, is required due to the presence of an archive in ParetoGP. At each generation we create a new archive by merging the old one of the previous generation with the new population, and selecting a fixed number of individuals located at the Pareto front in fitness-complexity space.

Since a *new* subset to evaluate the population is selected randomly at every generation, in principle, the fitness of the archive needs to be reevaluated using exactly the same subset. Smits and Vladislavleva (2006) considered this necessary, to make sure that we compare "apples with apples", when updating the archive for the next step. At each generation the extra computational effort is equal to the product of the archive size and the number of records in the current subset. This implies that the additional effort grows when the subset size increases during the run. For the case where the subset size increases from 10% to 100%, the archive re-evaluation step causes an increase of the total reference computational budget by 45%.

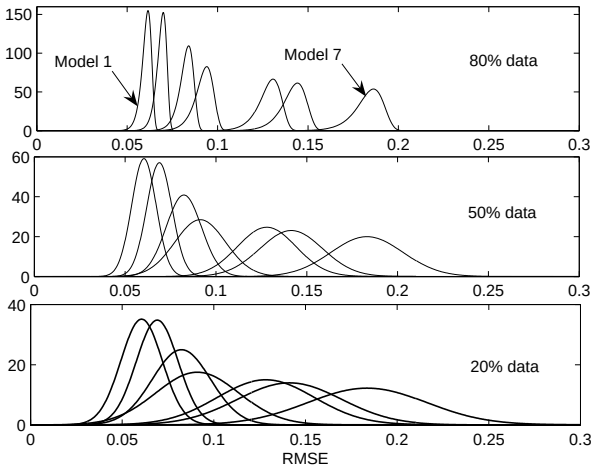


Figure 6.2: Approximated fitness distributions of the seven archive models from Figure 6.1 computed using random subsets of 20, 50, and 80% of the original data set for the Kotanchek problem.

The increase to 145% of computational budget is certainly not negligible, despite the fact that considerable improvement in the quality of solutions is obtained compared with the standard regression; see numerical experiments of (Smits and Vladislavleva, 2006).

6.3.2 Soft ordinal ParetoGP - better solutions with less effort

Given the additional effort required to re-evaluate the archive at every new generation, we hypothesize that re-evaluation of achive models on the same subset during updating the archive is not necessary to get a good enough ranking.

To observe the influence on the ranking of two individuals, we have to examine another distribution which is derived from a consideration of the difference in the estimated fitness of both individuals. This second distribution gives a direct estimate of the probability of having a wrong estimate of the rank of the two given individuals.

In Figure 6.3 the distributions of two different cardinal approaches to estimate the ranking of two individuals are modeled - taking the difference of fitness

estimates obtained on the same subset, and taking the difference of fitness estimates obtained using different subsets of a similar size.

Comparing fitness of models evaluated on the same subset (left-hand side of Figure 6.3) gives the most accurate estimates of the true difference in fitness, and correspondingly in the true ranking of the individuals. Surprisingly, the error in comparing fitness of models evaluated on different subsets (left-hand side of Figure 6.3) is still small enough to make reasonably accurate decisions on the ranking of individuals even for small subset sizes.

The essential elements of creating robust solutions are:

- **Abundant exploration**, caused by the use of larger population sizes and a softer selection process;

- **Sufficient exploitation**, caused by selecting potentially good parent models for further propagation. Despite the fact that the fidelity of the selection process is lower due to partial fitness evaluation, the exploitation of good models in the archive is still sufficient, due to the fact that winning models need to survive multiple low-fidelity screens to stay in the game and keep propagating. That being said, goal softening by means of partial fitness evaluations introduces new modes of exploitation. It allows bad-but-lucky models to propagate in the short term, but guarantees that truly good models (which are good on all data points) will survive multiple screens and keep propagating in the long term.

Since robust solutions of high quality are obtained when the balance between abundant exploration and sufficient exploitation is maintained, we hypothesize that allocating more computational budget to the initial part of the run (for exploration) at the expense of budget allocated to the end of run (for exploitation), may increase the speed-to-solution, provided that the exploitation stays sufficient at the end of the run. Such approach is discussed in the next section.

Goal softening comes from:

- the use of subsets instead of the entire data set;

- use of *random* subsets of a given size at every generation; (has a regularizing effect on the models by reducing the chances for over-fitting)

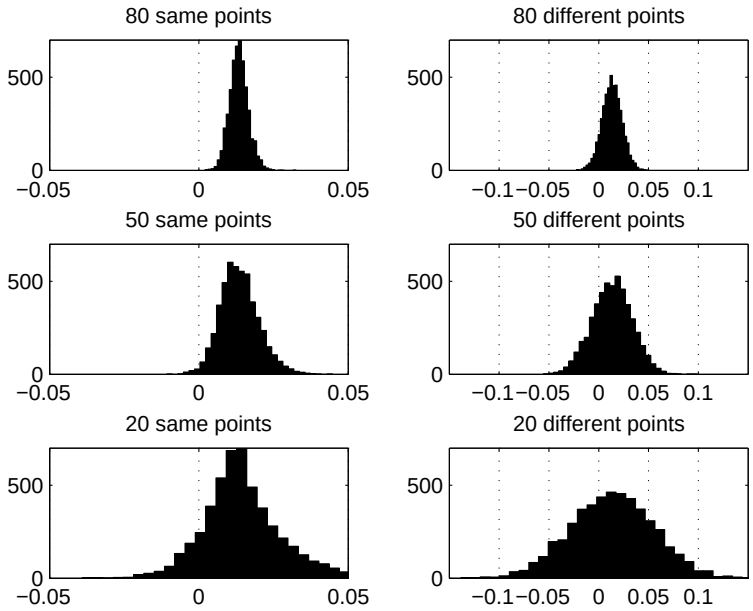


Figure 6.3: The histograms represent the difference in fitness values of the two archive models, calculated 10000 times on the same subset (the left-hand side), and different subsets of the same size (the right-hand side). The models used in this analysis are model 5 and model 6, selected in Figure 6.1. Notice, that the widths of the histograms on the left-hand side are much narrower than of the ones on the right-hand side. This indicates that a higher variance in the actual value of the difference in fitness values of the two models is introduced by evaluating them on different subsets. However, not the actual value of this difference is important. Only the sign matters to make a decision about the ranking. Thus, the fraction of the histogram area to the left of zero can be used to derive the probability of making an incorrect ranking. Notice, that even in the worst case (the lower-right plot) an error in ranking is made in only 32% of evaluations. This plot corresponds to evaluations on 20% of the original data - an evaluation scheme used only at the beginning of the evolution. Given our commitment to soften the selection goals at the first generation, and only gradually improve the screening towards the end of the evolution - a 30-40% chance to make a mistake in ranking becomes more than appropriate.

- not re-evaluating the archive models at every generation, and giving them a chance to stay in the archive for a while.

There is synergy among these principles for guiding the search for better and more robust solutions.

6.4 Case Studies

6.4.1 Goals of Experiments and test problems

Three test problems are used for regression experiments in this section:

1. **Salustowicz1d100 problem** is generated using the Salustowicz1d function (5.4); it contains 101 records, with inputs sampled uniformly from the range $[0, 10]$[4],

$$f_1(x) = x^3 \exp^{-x} \cos x \sin x (\sin^2 x \cos x - 1);$$

2. **Kotanchek problem**, drawn from (5.5), consists of 100 records, with inputs sampled randomly uniformly from the box $[0, 4] \times [0, 4]$,

$$f_2(x_1, x_2) = \frac{e^{-(x_2-1)^2}}{1.2 + (x_1 - 2.5)^2};$$

3. **Tower problem** is an industrial data set of gas chromatography measurements of the composition of a distillation tower. The underlying data set contains 5000 records with noise and 25 potential input variables.

Table 6.1 presents parameters for ParetoGP used in all experiments unless indicated otherwise.

The overall aim of all experiments was to get the highest quality results with the largest reproducibility and the lowest computational cost. In this chapter we focus on three case studies described below.

[4]In (Vladislavleva et al., 2007) this problem is referred to as **Maarten** problem, because the function and the data sets are kindly provided by Maarten Keijzer. With many thanks to Maarten, the decision is taken to synchronize the name of this problem with the name of the first author of a publication where the function first appeared (see (Keijzer, 2003; Salustowicz and Schmidhuber, 1997)).

Table 6.1: Parameter settings of reference ParetoGP runs. Note, that the other experiments described in Cases I-III have the same settings except for the population size, and the size of the data set used for fitness evaluations.

Number of independent runs	30
Number of cascades	10
Number of generations per cascade	25
Total number of generations	250 (500 in PGPA)
Population size	100 (200 in PGPB)
Archive size	100
Run Performance measure	Area% under Pareto front
Accuracy measure	$1 - NMSE$ (smaller values preferred)
Complexity measure	Expressional
Population tournament size	5
Archive tournament size	3
Crossover rate	0.95
Mutation rate	0.05
Rate of mutation on terminals	0.3
Function set 1(**Kotanchek, Tower**)	$+,-,*,/,e^x,e^{-x}$, x^{real}, $x + real$,$x \cdot real$
Function set 2 (**Salustowicz1d100**)	Function set 1, $\sin x$, $\cos x$

CASE I

The first experiment is an extension of a setup, discussed in (Smits and Vladislavleva, 2006). It is based on partial fitness evaluation of GP individuals on a *random* subsets of the original data set. Now, the selection of subsets is performed randomly and uniformly from the collection of the available data records. During the run the subset size is gradually increased, and the population size is gradually decreased. As a result, a better exploration is pursued at the beginning of the evolution when more individuals are evaluated at a time, albeit more coarsely, and exploitation is improved towards the end of the run, when fitness evaluations are refined by adding more data points. Five different experiments are performed for various starting subset sizes of {10%, 20%, 40%, 60%, 80%} of the original data size.

Unlike (Smits and Vladislavleva, 2006), the number of function evaluations per generation is constant. Therefore, when the scheme of increasing the data subset size is chosen, and the number of records used for each generation is

defined, the population size for a given generation is obtained by dividing the number of function evaluations (the budget) per generation by the number of records in a subset. All experiments were repeated at least 50 times to get reliable statistics.

CASE II

In a second set of experiments we are repeating the set up described in Case I with one small modification. We are no longer re-evaluating individuals in the archive at every generation, but only at every tenth generation. The computational budget assigned to these archive re-evaluations is cut by 90% compared with Case I. This introduces further softening of the selection process, since archive individuals that accidentally obtain (unrealistically) high fitness values, can still survive in the archive for up to ten generations, and, hence, compete and propagate their features to their offspring. As noted in the previous section, the error in ranking caused by this approach are relatively small, and should still produce solutions comparable with the solutions of Case I.

CASE III

In the previous two cases the computational budget per generation was kept constant. In the third set of experiments we examine the effect of re-distributing the total computational budget in such a way that we spend 50% more function evaluations at the first generation, gradually decrease the budget over the run, and end up with spending 50% less at the last generation, compared with the corresponding runs of Case I and II. The total budget per run, however, does not change. The intention is to have even more exploration in the beginning of the run, at the expense of exploitation near the end of the run. The scheme for archive re-evaluations is the same as in Case II (once every ten generations).

6.4.2 Empirical results

The simulation results for the three cases are summarized in Table 6.2. We used the following criteria for estimating the performance of different experiments: the median and the interquartile range (IQR) of the percentage of the area under the Pareto front for 50 independent replicates, and the total number of function

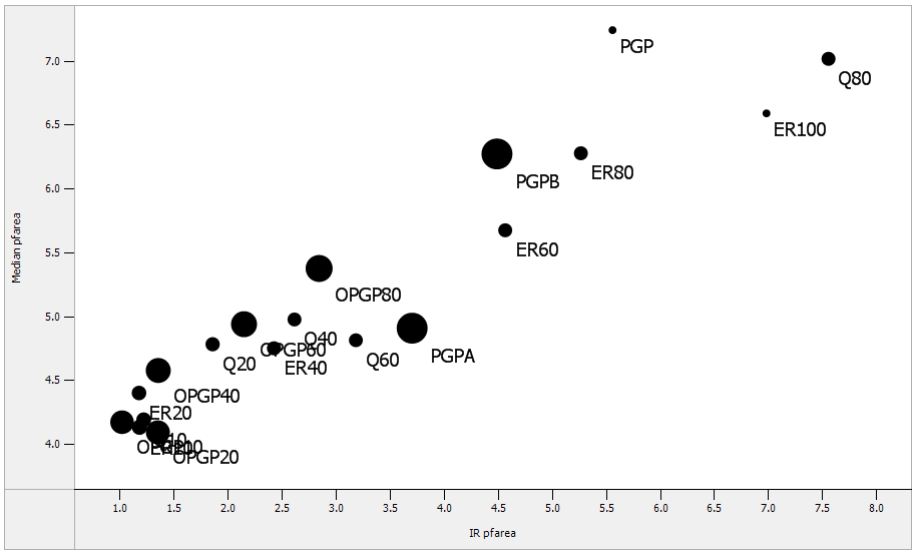


Figure 6.4: Summary of all results for the Salustowicz1d100 problem. The bubbles represent the results of the different experiments in Cases I-III in terms of the median performance measure over 50 independent runs (y-axis), interquartile range of the performance measure (x-axis), and the normalized total number of function evaluations per run (the size of the bubbles). Smallest bubbles located at the zero region are preferred. Same labels as in Table 6.2 are used. See a magnified area of interest in Figure 6.5.

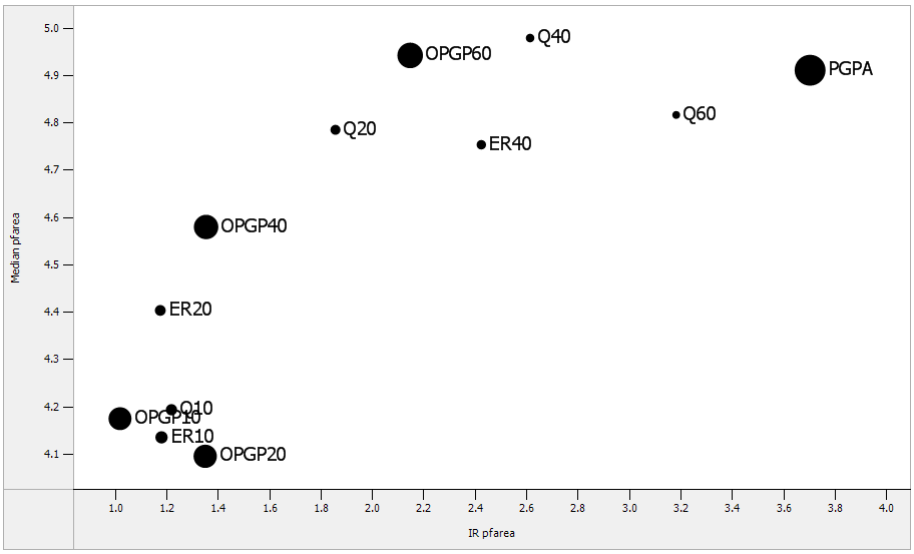
evaluations per run normalized by the size of the training data set. For each criterion, lower values indicate better performance.

The results in Table 6.2 are also displayed as bubble charts in Figures 6.4, 6.5, 6.6, and 6.7. The size of the bubbles in these charts is proportional to the budget that was spent for a particular experiment.

Examining the results for **Case I**, we observe a clear improvement of the median fitness as well as the IQR for smaller starting subsets. The surprising fact is that even for very small datasets (like Salustowicz1d100 and Kotanchek) the optimal strategy is to start with a low subset size of 10% to 20%. This is most probably related to the low dimensionality of these problems. Note, that the runs starting with the smaller subset sizes also consume less computational budget because the archive reevaluations are cheaper. The ordinal runs starting with small subset sizes (OPGP10 and OPGP20) considerably outperform the reference ParetoGP runs with a constant population size of 100 (PGP), and even those with twice the number of generations (PGPA) or twice the population size

Table 6.2: Empirical results for three test problems. The table represents the results of different experiments in terms of the median and the interquartile range of our performance measure over 50 independent runs and the normalized total number of function evaluations per run (cpu budget). The performance measure is the percentage of the area under the Pareto front of the archive solutions in complexity vs. fitness space. For all three quality characteristics - median and the IQR of the Pareto front area percentage, and the cpu budget - smaller values are preferred. There are three types of standard reference runs: PGP (ParetoGP) - with the population size of 100, and 250 generations, PGPA with twice the number of generations, and PGPB with twice the population size compared with the PGP runs.

Experiment	**Salustowicz1d100**		**Kotanchek**		**Tower**		Budget
	$\tilde{\mu}$	IQR	$\tilde{\mu}$	IQR	$\tilde{\mu}$	IQR	per run
Reference							
PGP	1.81	1.39	2.20	0.735	1.55	0.325	250000
PGPA	1.23	0.93	2.05	0.786	1.39	0.334	500000
PGPB	1.57	1.12	2.06	0.630	1.41	0.334	500000
Case I							
OPGP10	1.04	0.26	1.86	0.662	1.33	0.209	367169
OPGP20	1.02	0.34	2.02	0.633	1.37	0.368	372486
OPGP40	1.15	0.34	2.10	0.764	1.39	0.300	387703
OPGP60	1.24	0.54	2.18	0.831	1.57	0.362	404590
OPGP80	1.34	0.71	2.30	1.002	1.46	0.422	422122
Case II							
Q10	1.05	0.31	1.93	0.431	1.32	0.250	273119
Q20	1.20	0.46	1.95	0.382	1.34	0.253	269886
Q40	1.24	0.65	2.18	0.652	1.41	0.285	268003
Q60	1.20	0.80	2.23	0.630	1.56	0.405	267790
Q80	1.76	1.89	2.14	0.710	1.44	0.380	268222
Case III							
ER10	1.03	0.30	2.00	0.624	1.52	0.387	276610
ER20	1.10	0.29	1.92	0.571	1.35	0.296	271850
ER40	1.19	0.60	2.00	0.527	1.35	0.333	268876
ER60	1.42	1.14	2.12	0.520	1.39	0.287	268201
ER80	1.57	1.32	2.14	0.809	1.42	0.331	268379
ER100	1.65	1.75	2.32	0.802	1.64	0.327	250000



(a) Magnified area of interest

Figure 6.5: Summary of results for the Salustowicz1d100 problem. Magnified area of interest. The bubbles represent the results of the different experiments in Cases I-III in terms of the median performance measure over 50 independent runs (y-axis), interquartile range of the performance measure (x-axis), and the normalized total number of function evaluations per run (the size of the bubbles). Smallest bubbles located at the zero region are preferred. Same labels as in Table 6.2 are used. PGP - reference ParetoGP runs, OPGP - ordinal ParetoGP runs of Case I, Q - quick ordinal runs of Case II with less frequent archive re-evaluations, ER - exploratory runs of Case III with re-allocated computational budget. The numbers in the labels indicate the starting percentage of the subset size. The area of interest is the lower-left corner of the graph. The most successful experiments for the Salustowicz1d100 problem are OPGP10, Q10, ER10, OPGP20, which again confirms that starting from small subsets leads to solutions superior in terms of best median fitness and consistency of independent runs. Note the substantial budget savings in Q10 and ER10, which correspond to Case II, and Case III, respectively.

(PGPB).

In **Case II**, we examine the effect of not re-evaluating the archive individuals at every generation but only once every ten generations. The bubble charts show only a slight deterioration in the results for the median fitness and the IQR, compared with the ordinal runs from Case I. Considering the budget, these experiments are nevertheless very competitive. The advantage is that the computational budget is now very close to the original budget of the PGP reference run.

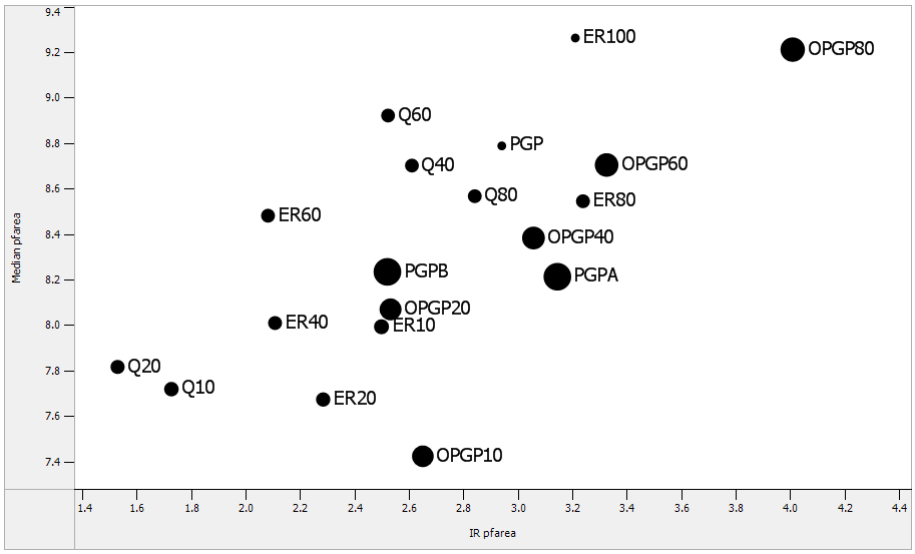


Figure 6.6: Summary of results for the Kotanchek problem. The bubbles represent the results of the different experiments in Cases 1-3 in terms of the median performance measure over 50 independent runs (y-axis), interquartile range of the performance measure (x-axis), and the normalized total number of function evaluations per run (the size of the bubbles). We use the same labels as in Table 6.2. PGP - reference ParetoGP runs, OPGP - ordinal ParetoGP runs of Case I, Q - quick ordinal runs of Case II with less frequent archive re-evaluations, ER - exploratory runs of Case III with re-allocated computational budget.The numbers in the labels indicate the starting level of the subset size. The area of interest is the lower-left corner of the graph. Note, that the most successful experiments are Q20,Q10, ER20, OPGP10, which confirms that starting from small subsets leads to solutions superior in terms of best median fitness and consistency of independent runs.

In **Case III** the aim was to examine whether there was any benefit in relocating a part of the computational budget from the end to the beginning of a run. The results are comparable with the results of Case II, however there is a clear deterioration in reproducibility of the independent replicates. We speculate, that by relocating 25% of the computational budget from the second half of the run to the beginning, we actually assign too much weight to exploration and not enough to exploitation. While a thorough analysis of other budget distribution schemes is required, the provisional conclusion is that a constant computational budget per generation is a good default strategy.

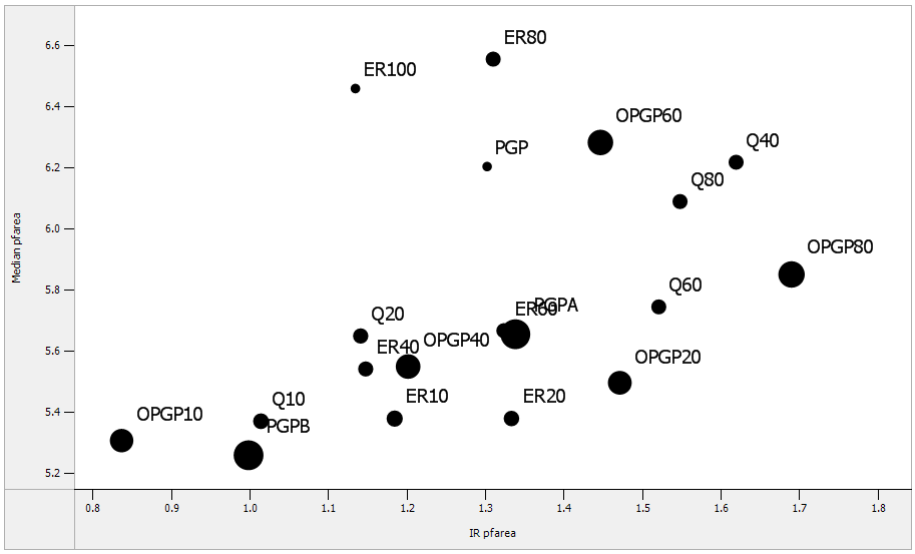


Figure 6.7: Summary of results for the Tower problem. The most successful experiments for the Tower problem are OPGP10, Q10, and PGPB. The first two are no surprise, we observe the same results for other test problems. The PGPB corresponds to the reference ParetoGP runs that uses twice the population size for the same number of generations. The high quality of these results is caused by the nature of the tower problem. This real industrial problem is more challenging with respect to the variable selection (25 candidate inputs). Besides, the exact underlying input - output relationship may not exist. It is no surprise that PGPB runs with a bigger population size and hence better exploration produce better solutions than exploitative PGPA runs with more generations. It is important to note, that this improvement requires a higher computational budget.

6.5 Summary

This chapter confirms and refines the findings of (Smits and Vladislavleva, 2006) that the use of goal softening consistently generates results that are superior both in terms of median fitness, consistency between independent runs, and required computational budget. It can be concluded that starting with subset sizes of only 10% to 20% generates optimal results for all three test problems.

More analysis is needed to find appropriate default settings for the starting subset size. If the proposed strategy works well on small subset sizes of 10% of the original size, smaller sizes may be beneficial since they allow even larger starting population sizes for the same computational budget, and, hence, better exploration.

Experiments also confirm that there is no need to re-evaluate the archive individuals at every generation, keeping essentially the same quality of results at the lower computational budget. Finally, the experiments of this chapter demonstrate that keeping the number of function evaluations per generation constant over the run, is a good default setting and seems to provide the necessary balance between exploration and exploitation in soft ordinal Pareto GP.

In some way, the goal softening by soft ordinal Pareto GP, as many other subset selection schemes, or niching mechanisms, like age-layered population structures (Hornby, 2006), or Hierarchical Fair Competition (Hu et al., 2005) introduce new modes to exploitation - they allow bad-but-lucky models to propagate in the short term, but guarantee that truly good models, that are *robust* and good on all data points, will survive multiple screens and proliferate in the long term.

Improvement in the performance of Soft Ordinal Pareto GP over the standard Pareto GP can be explained by a proper balance of abundant exploration and sufficient exploitation. A softer selection process with partial fitness evaluations allows using larger population sizes for better exploration. The fact that 'good' individuals have to survive multiple low-fidelity fitness evaluations on different subsets to stay in the archive and keep propagating allows sufficient exploitation of 'good' genetic material.

Two major questions remain. The use of stochastic subset selection and changing the subsets at each generation seems to pursue additional regularization among regression models, because they need to survive multiple screens on different subsets to stay in the archive. Lower values of the Pareto front area percentage corresponding to the solutions of the soft ordinal approach support the hypothesis of (Banzhaf et al., 1998) about regularization effects of stochastic sampling, but raises a question: Is randomness necessary to achieve robustness? Giving up randomness may contribute to active information re-use, at least for models that stay in the archive and do not need to be fully reevaluated if the training subset does not change completely.

The second question is whether the alternative strategies for increasing the subset size in a non-linear fashion can provide further enhancement in the quality of final solutions. The issues of non-stochastic subset selection, non-linear subset size increase, and possible information reuse are addressed in the next chapter.

7

Model Evaluation through Incremental Evolutions on Essential Data Records

This chapter introduces a novel strategy for a *quick* generation of compact and accurate regression models on multi-dimensional input-output data via Pareto genetic programming. The ESSENCE algorithm evaluates *more* potential solutions on *smaller* subsets of data, starting with 'the most important' data records.

This algorithm ranks the training set in the order of decreasing importance and partitions it into *nested* subsets of *non-linearly increasing size*. The speed at which the size of the training subset is increased depends on the balancedness and compressibility of the input-output data, and is determined by the cumulative information content of the data.

Since the training subsets are nested in the ESSENCE algorithm, and only slightly change from generation to generation due to the addition of new points, further savings of the computational budget can be achieved, if intermediate fitness values of persistent individuals are archived and updated when new data are added to the training subset.

7.1 Motivation

The previous chapter presented a soft approach to symbolic regression via archive-based Pareto genetic programming in the spirit of ordinal optimization. It pursues abundant exploration by coarse evaluation of *more* potential solutions

at the beginning of the run and maintains sufficient exploitation by refining the evaluation process in the course of a run. The fidelity of fitness evaluations depends upon the size of the training subsets, which *increase linearly* during the run. The stochastic subset selection in the presented strategy enhances the robustness of the solutions obtained due to additional regularization.

This chapter presents another method of reducing the computational effort of an evolutionary algorithm, aimed at improving the effectiveness of the search by quickly zooming into the promising areas of the search space.

In Figure 7.1 different schemes for classical and ordinal approach to evolutionary algorithms are plotted. The top plots represent the percentage of the size of the training set per generation. The bottom plots present the corresponding size of the population at each generation. Plots in the first column refer to the standard regression setting, when a given set of individuals is evaluated on all fitness cases at each generation. The plots in the second column illustrate an example of the ordinal approach, when the size of the training set linearly increases from 20% to 100% of the original size over the first 95% of generations, and at the last 5% of generations individuals are evaluated on the total training set. Due to partial fitness evaluation, the population size can be effectively increased up to 500% of the original size, provided that the computational budget per generation is constant and equals the total number of function evaluations in standard regression setting.

The experiments of the previous chapter performed on three test problems indicated that with the same computational budget a significant improvement can be achieved through an ordinal approach to symbolic regression over the standard Pareto GP approach with respect to the quality of solutions and reproducibility of independent runs. Starting with a subset size of 10% to 20% of the training set provided the best results for all three test problems.

Changing subsets at each generation in the soft ordinal Pareto GP seemed to be necessary to obtain a significant improvement in the results, since keeping the same subset for several generations and increasing it in a step-wise fashion did not produce significant improvements; see (Smits and Vladislavleva, 2006). This raised the question whether the training subsets changing at each generation and drawn randomly from the training set are necessary to obtain the improvement in reproducibility and quality of solutions. The second question was about the alternatives for a linear increase of the training subset size.

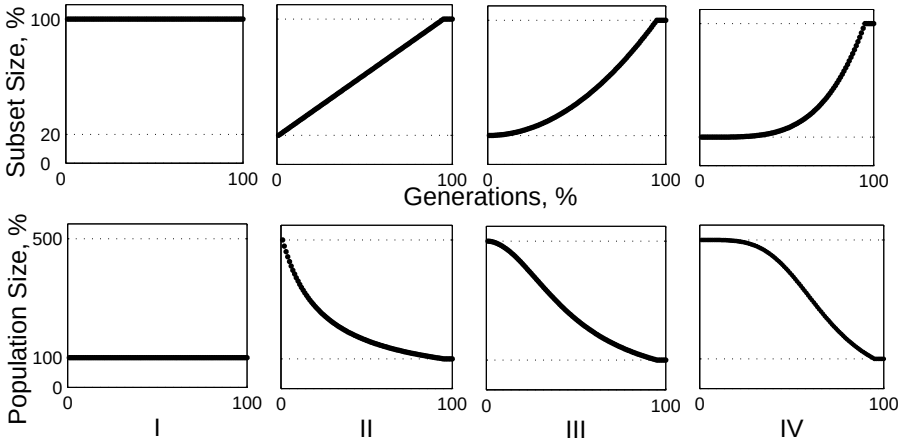


Figure 7.1: Schemes for partial fitness evaluation with the fixed budget allocation. The four columns the figure represent four various modeling schemes with partial fitness evaluations on subsets of the given data set. Column I represents a reference GP run where a population of a fixed size is evaluated on all records of the data set over all generations. In column II the size of the subsets used for partial fitness evaluations are increasing linearly from 20% to 100% over the first 95% of generations, and over the last 5% of generations the classical scheme is used. The population size per generation is computed according to the formula $100/subsetSize \times 100\%$. Columns III and IV are the examples of the schemes for non-linear increase of the subset size, that implies a gain in the cumulative population sizes over the run, and hence further improves exploration.

From the plots in the columns three and four of Figure 7.1 we observe that if the size of the training subset is increased slower than linearly, a larger exploration can be achieved with bigger population sizes over most of the generations. Of course, the improvement of individuals will take place only if the balance between exploration and exploitation is maintained. The hypothesis is that if subsets contain a *sufficient amount of information* about the underlying response surface at all times, the balance between exploration and exploitation will be preserved even if longer training is applied to small subsets, and the subset size increases slowly (e.g., as in columns three or four of Figure 7.1).

Chapter 2 of this thesis presents a technique for weighing and ranking the records in the input-output training set in the order of decreasing importance. The results of Chapter 2 suggests that the proposed SMITS procedure when applied to one of the four weighing functionals, can be successfully used for

compression of the training set to smaller subsets containing a similar amount of information about the data.

The main property of the training set ranked with the SMITS procedure is the nested balancedness of its subsets. In other words, the first m records of the ranked set are sufficiently 'space-filling' for each integer size m, $m < N$, $m > k$, where N is the number of records in the training set, and k is the neighborhood size used for weighing the data[1]. This interesting property calls for an application of the ordinal approach presented in the previous chapter to the *nested* subsets of the training set ranked with the SMITS procedure.

If the subsets of increasing size in the ordinal approach to genetic programming are nested, we can extensively re-use information about persistent individuals and building blocks from previous generations. Besides, nested subsets slowly changing over several generations would allow additional racing of individuals as in (Teller and Andre, 1997). On the other hand, the use of nested subsets over a GP run may increase the complexity of individuals, which would be the opposite effect compared with the regularization observed for the use of random subsets changing at each generation.

The first goal of this chapter is to find out whether the use of nested subsets can outperform the use of random subsets in soft ordinal Pareto GP.

The second goal of this chapter is to suggest a sensible procedure for a non-linear increase of the subset size such that it improves the quality of GP solutions for both random and nested sub-sampling.

7.2 ESSENCE algorithm

This section introduces a novel heuristic to define the scheme for increasing the subset size, which is based on the cumulative information content of the training set ranked in the order of decreasing importance of records. The new scheme of the non-linear subset size increase is designed for incremental modeling of nested subsets of the ranked training set, but can be also used with random subsets, as we show in the Case Studies in the next section.

The new algorithm employs methods of data weighing and ranking presented

[1] In more detail, the subset of the first m records of the ranked set has the maximal cumulative weight compared with other subsets of m records, where the weight is defined according to a selected weight functional

in Chapter 2: 1) Weighing the data with surrounding and non-linearity weights (with respect to surrounding by k nearest-in-the-input space neighbors, and with respect to the deviation from the least-squares regression hyper-plane approximating the k nearest-in-the-input-space neighbors); 2) Ranking the data in the order of decreasing importance with the simple multi-dimensional iterative technique for sub-sampling (SMITS) applied to a selected weight functional; 3) Estimating the cumulative information content (CIC) of the data set ranked with the SMITS procedure using a selected weight functional.

The SMITS procedure ranks the input-output data set of size N in such a way that for an arbitrary integer number m, $k < m < N$, the first m records of the sorted set always form a sufficiently balanced subset. The 'balancedness' is interpreted in terms of the selected balancing, i.e. weighing functional. We suggest to use the SMITS procedure with four weighing functionals for sensible compression of data to balanced nested subsets of arbitrary size.

The first idea of the new algorithm consist in using the nested subsets of the first m records of the data set ranked by the SMITS procedure for incremental modeling. The ordinal approach of modeling small subsets with big populations and gradually increasing the size of the subsets while decreasing the size of the populations can be applied to the SMITS-based ranking of the training set. This approach may reinforce discovery and exploitation of good genetic material from the very beginning of the evolution, because the small data sets obtained as described above will by design contain maximal information content compared with other subsets of similar sizes. The exploration of alternative solutions will still be maximized, since small training subsets will allow large population sizes within a fixed computational budget.

Obviously, the speed of increasing the subset size should depend on the structure of the total data set and on the amount of information contained in the starting subset. If the size of the training subset grows too quickly, the additional exploration guaranteed by the ordinal approach will quickly deteriorate, because the population sizes will have to be decreased to keep the budget constant. If the size of the training subset grows too slowly (e.g., does not increase at all), the exploitation may deteriorate at the expense of over-exploration. Besides,the performance of final solutions will be too sensitive to the starting subset size, to the imbalancedness of the original data set, and to the success of the exploration stage.

The second idea of the new algorithm is to use the shape of the cumulative information content of the ranked data set to determine the speed, at which nested subsets increase in size. For example, imagine, that by analogy with the soft ordinal approach to symbolic regression, the user decides to start modeling with a training subset of 10% of original size and wants to incrementally add points until 100% of data is added to the system by 95% of the total number of iterations (generations). Instead of linearly increasing the size of a training subset from 10% to 100% over 95% of generations, we suggest to linearly increase the information content of the training subset from 10% to 100% over the same number of generations. In general, this approach implies a non-linear increase of the size of the subset.

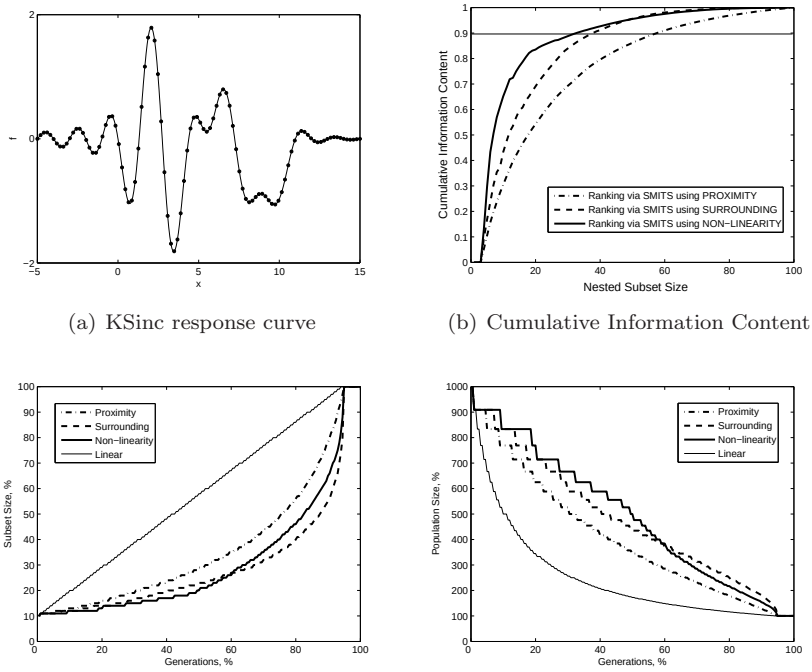
In Figure 7.2 we illustrate the procedure on the Ksinc data from Chapter 2. Figure 7.2(c) presents schemes of a non-linear addition of new points to the training subset set such that the cumulative information content of the used subset (obtained with a corresponding weight) grows linearly per generation.

To obtain the scheme plotted in Figure 7.2(c) the user needs to find the value $CIC_{0.1}$ of the cumulative information content corresponding to the first 10% of records of the ranked training set, and then quantize the CIC curve on the interval $[CIC_{0.1}, 1]$ by the number of levels corresponding to 95% of planned generations. The subset size, corresponding to the CIC at level i will determine the size of the training subset at generation $\frac{i}{100} totalGenerations$.

The scheme for decreasing the population size follows from the scheme of increasing the subset size; see Figure 7.2(d). Since all comparisons in this chapter are based on a fixed computational budget, the population size corresponding to the training subset of r% records ($r \in (0, 1]$) is equal to $100/\frac{r}{100}$% of the original population size.

The suggested procedure of applying symbolic regression to the training set ranked with the SMITS procedure with incremental addition of new points to the training subset such that its cumulative information content increases linearly over the evolutionary run, will be referred to as the ESSENCE algorithm, because it is expected to perform an Effective Search Space Exploration (via Nested Content-based Evolutions).

Due to the fact that the size of the training subset in the ESSENCE algorithm may change every generation, the archive in the ParetoGP system still needs to be re-evaluated on a training subset, common with the population's training



(a) KSinc response curve

(b) Cumulative Information Content

(c) Subset Size per Generation

(d) Population Size per Generation

Figure 7.2: Cumulative Information Content guides the speed of increasing the training subset size for incremental modeling. The first plot represents the Ksinc data set of 100 points and the tru response curve, defined by Eq. (2.13). The second plot represents the cumulative information content of the first m records of the **KSinc** data set ranked with the SMITS procedure for the proximity, surrounding, and non-linearity weight on two nearest-in-the-input-space neighbors. The second and the third plots represent the scheme for subset size increase and the corresponding scheme for population size decrease (in percentages of the original sizes) for an incremental modeling run, where training data consists of nested subsets of the ranked set and is supplied in a non-linear fashion such, that the cumulative information content increases linearly per generation. The population is computed as $100/subsetSize$ to keep the computational budget constant per generation. The thin black line in Figure 7.2(c) corresponds to the linear increase of the subset size used in (Smits and Vladislavleva, 2006) for random subsets. In Figure 7.2(d) the difference in the areas under the curves quantifies the gain in the exploration due to bigger population sizes.

subset once in a while. We prefer to do it at the end of each cascade, as in the soft ordinal ParetoGP approach, when the population is re-initialized.

In the next section the ESSENCE algorithm via ParetoGP is compared with the with the soft ordinal ParetoGP presented in the previous chapter.

7.3 Case Studies

7.3.1 Goals of experiments

The goal of the experiments in this chapter is two-fold. First, we want to compare the effectiveness of regression on *random* subsets with the use of *nested* subsets obtained from a good ranking of the training set. Second, we want to study whether the ESSENCE algorithm on nested subsets with new points added incrementally in a non-linear fashion produces better results than the soft ordinal approach on random subsets with sizes increasing linearly over a GP run.

Based on the conclusions of experiments and demonstrations of Chapters 2 and 5, two weighing functionals can be selected for ranking training sets with the SMITS procedure, the surrounding and non-linearity weights. The default number of nearest-in-the-input-space neighbors, suggested in Chapters 2 and 5 is the number of *significant* input dimensions plus one.

Chapter 5 indicated that weighing and ranking the data with non-linearity weights seems to be the best general purpose approach (for a reasonable number of dimensions), but it can be inferior to the use of the surrounding weight in cases of extreme imbalancedness of the data, when the distances to k nearest -in -the -input space neighbors differ significantly. Such situations would imply that the deviation from the hyper-plane passing through those neighbors looses the meaning of local non-linearity of the response surface at a given point. In the experiments of this chapter we intend to further analyze how the difference in the SMITS-based rankings of data with the non-linearity and the surrounding weight is affecting the quality of GP solutions in the incremental modeling framework [2].

A comprehensive comparison of the standard ParetoGP reference runs with the soft ordinal runs and the ESSENCE runs is performed according to the scheme given in Figure 7.3.

[2]We expect the ESSENCE algorithm based on the surrounding weight to outperform the ESSENCE algorithm based on the non-linearity weight at least on the **KotanchekImbalanced**

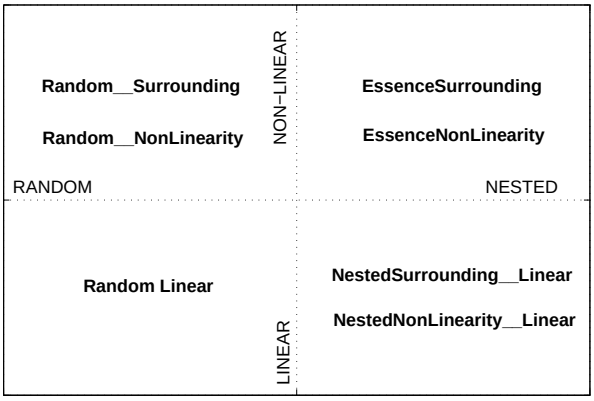


Figure 7.3: Types of Experiments. The horizontal axis divides the experiments by subset selection scheme, the vertical axis divides the experiments by the speed of subset size increase.

Below we provide notations and explanation of all experimental schemes and emphasize the main comparison groups with an asterisk (*). The first word in the experiment name (except for Reference runs) stands for a subsetting scheme - Random - codes random subsets, Essence or Nested codes nested subsets. The second word in the experiment name for Nested and Essence experiments codes a procedure for determining the ranking of points: NonLinearity corresponds to the non-linearity weight, Surrounding corresponds to surrounding weight. 'Rand' in the NestedRand_NonLinearity experiment stands for a ranking obtained with a random transposition of records in the training set.

The word after the underscore symbol in the experiment name stands for the speed, at which the subset size is increased. _ Linear codes linear increase. _NonLinearity codes the speed determined by the cumulative information content of the data, ranked with the SMITS procedure using non-linearity weight. _Surrounding codes a non-linear speed determined by the cumulative information content of the data, ranked with the SMITS procedure using surrounding weight. The speed of increasing the subset size in the Essence experiments is determined by the weight functional used for the SMITS-based ranking (i.e. by the shape of the corresponding cumulative information content). Summarizing, we have the

and the **Tower** problems.

following experiments:

- **REFERENCE*** denotes the standard reference ParetoGP runs with settings given in Table 7.1.
- **Random_Linear*** denotes the soft ordinal GP runs on subsets of the training set drawn *randomly* at each generation, with the size of the subsets increasing *linearly* from 10% to 100% in the first 95% of the total number of generations.
- **EssenceNonLinearity*** stands for the ESSENCE GP runs on the nested subsets of the training set ranked with the SMITS procedure based on the non-linearity weight, with the subset size increasing in a non-linear fashion based on the cumulative information content of the ranked training set used.
- **EssenceSurrounding*** - stands for the ESSENCE GP runs on the *nested* subsets of the training set ranked with the SMITS procedure based on the *surrounding* weight, with the subset size increasing in a non-linear fashion based on the cumulative information content of the ranked training set used.
- **Random_NonLinearity** and **Random_Surrounding** stand for the soft ordinal GP runs on subsets drawn *randomly* at each generation, with the size increasing in a non-linear fashion, based on the cumulative information content of the nested subsets obtained with the SMITS procedure based on the *non-linearity* and the *surrounding* weights respectively. This means that the speed at which the subsets size is increased is the same as in the Essence runs, but the subsets are not nested, but drawn randomly from the entire subset. Note, that the non-linear speed of adding new points is based on the assumption that the subset of the first m records is as balanced as possible with respect to the selected weight functional. This implies that it will contain the maximal amount of information compared with other subsets of size m, and particularly with a subset of m records drawn randomly from the training set. The difference in information contents will only increase for the data sets that are not balanced.
- **NestedNonLinearity_Linear** and **NestedSurrounding_Linear** denote the GP runs on nested subsets of the training set, ranked with a SMITS

procedure using the *non-linearity* and the *surrounding* weights, where the size of the subset increases *linearly* from 10% to 100% of the original size in the first 95% of generations.

- **NestedRand_Nonlinearity** stands for the GP runs on nested subsets of the training set, with records permuted in a random order, and the size of the subsets increasing in a non-linear fashion from 10% to 100% of the original size according to the shape of the cumulative information content of the set ranked with the SMITS procedure using the non-linearity weights. We perform this experiment to find out whether the special order of data induced by the SMITS procedure is essential to obtain the improved results.

7.3.2 Test problems and GP settings

Experiments mentioned above are performed on three test regression problems.

1) **KotanchekImbalanced problem** (see Chapter 5) consists of 100 points sampled from the Kotanchek function (Eq. 5.5) on the interval $[0, 4] \times [0, 4]$ in a special non-uniform way:

$$f_1(x_1, x_2) = \frac{e^{-(x_1-1)^2}}{1.2 + (x_2 - 2.5)^2}.$$

At most one half of the records of the KotanchekImbalanced data captures the non-linearity of the response surface, see Figure 7.4(a). The test set for the KotanchekImbalanced problem consists of 1681 points and forms a uniform mesh on the input interval $[0, 4] \times [0, 4]$.

2) **Salustowicz2dBioN problem** is sampled from the Salustowicz2d response surface (see Eq. (5.6)):

$$f_2(x_1, x_2) = (x_2 - 5)x^3 e^{-x} \cos x \sin x(\sin^2 x \cos x - 1).$$

The input set comes from the study of the relation of plants' feeding systems to the amount of the floating sludge for the Biox water purification plant. While the input points of the Salustowicz2dBioN problem correspond to an undesigned set of 3931 measurements of two Biox-related variables scaled to the interval $[0, 10] \times [-5, 15]$, the response is computed from equation (5.6), see Figure 7.5(a). The test set for the Salustowicz2dBioN problem contains 4489 records, and forms a

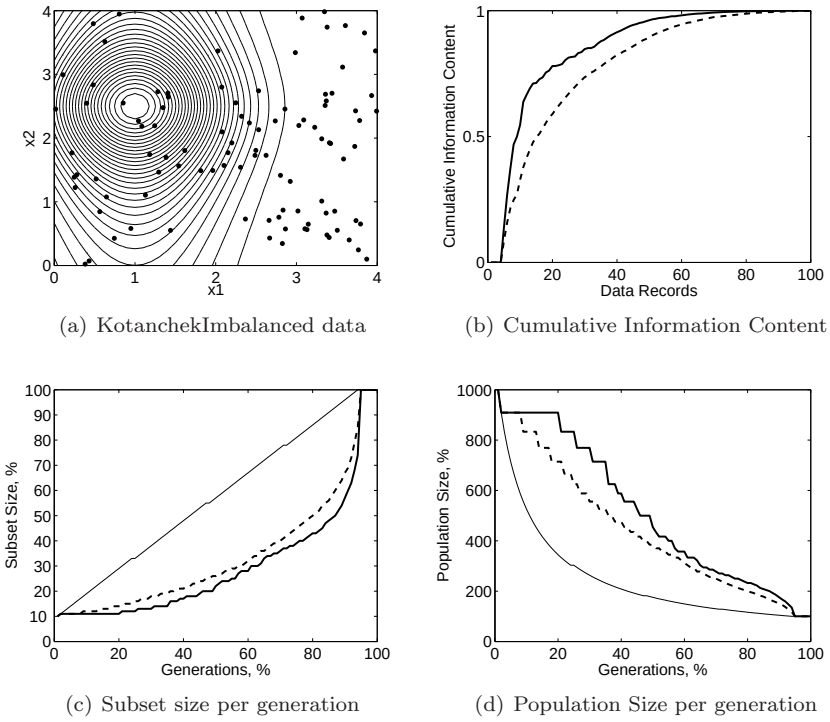
uniform mesh on the interval $[0, 10] \times [-5, 15]$. Note that the input interval for the Salustowicz2dBioN problem is different from the interval of the Salustowicz2dBio problem, analyzed in Chapter 5. The latter used the Biox input set scaled to $[0, 15] \times [-2, 13]$.

3) **Tower problem** comes from an industrial application and consists of 5000 records and 25 potential input variables; see Smits and Vladislavleva (2006) and Chapter 5. It represents gas chromatography measurements of the composition of a distillation tower. We plot the output of the tower problem in Figure 7.6(a). The test data for the Tower problem is the same as the training data.

The Tower data set went through a preprocessing stage that is worth describing. This real-life data contains 25 input variables, many of which may be insignificant in the input-output relationship. To apply the ESSENCE algorithm to this problem, we needed to rank the data with the non-linearity and the surrounding weights in the subspace of significant dimensions. We therefore performed the screening runs of 20 generations to obtain the ranking of driving variables, as described in Smits et al. (2005). The screening runs resulted in the following subset of significant variables: x_1, x_4, x_6, x_{12}, and x_{23}.

We performed the SMITS ranking of the Tower data in the five-dimensional input space of significant variables. Initially the entire set of 5000 records was weighted with the non-linearity and surrounding weights. After the first weighing we observed that the weights of two points (both non-linearity and surrounding) exceeded the weights of other points 250 and 300 times. These high weights of two records clearly indicated outliers, which were removed from the training set. The resulting set of 4998 records was then ranked by the SMITS procedure in the five-dimensional subspace of inputs with a fractional distance metric $\|\cdot\|_{1/6}$ (see Chapter 2 for fractional distances). The cumulative information contents and schemes for subset size increase plotted in Figures 7.6(b) and 7.6(c) are also computed in the space of significant inputs and response.

The rankings obtained from the SMITS procedure using non-linearity and surrounding weights were used to rank the total Tower training set of 25 input variables. This process resulted in the same number of variables used in all experiments including the ESSENCE experiments. To ensure fair comparisons, the ESSENCE experiments and experiments on nested subsets were performed on 180 generations only, compared with 200 generations of the reference runs since the screening runs aimed at variable identification are 20 generations long.



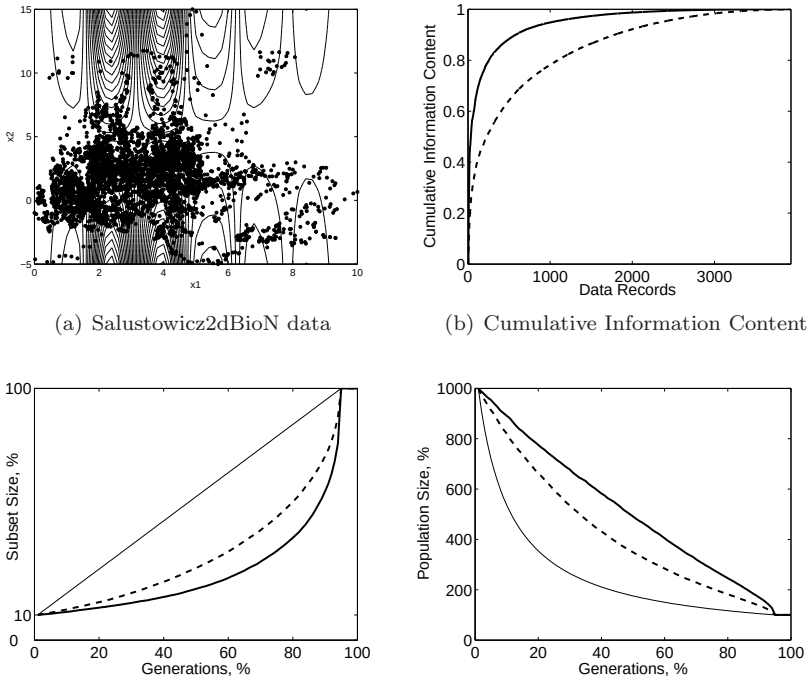
(a) KotanchekImbalanced data

(b) Cumulative Information Content

(c) Subset size per generation

(d) Population Size per generation

Figure 7.4: KotanchekImbalanced data. The Figure represents the contour plot of the two-dimensional KotanchekImbalanced data with 100 points sampled non-uniformly from the input interval $[0, 4] \times [0, 4]$. At least a half of the data records does not capture the non-linearity of the response surface. Plot (b) represents the Cumulative Information Contents of the two rankings of the KotanchekImbalanced data obtained with the SMITS procedure with the non-linearity weights (thick black line) and the surrounding weights (dashed line). Plot (c) illustrates the scheme of addition of new points for the ESSENCE algorithm, calculated based on the CICs of the plot (b) and on the intention to start modeling with subsets of 10% of the original size and to increase the subset size over the first 95% of generations.



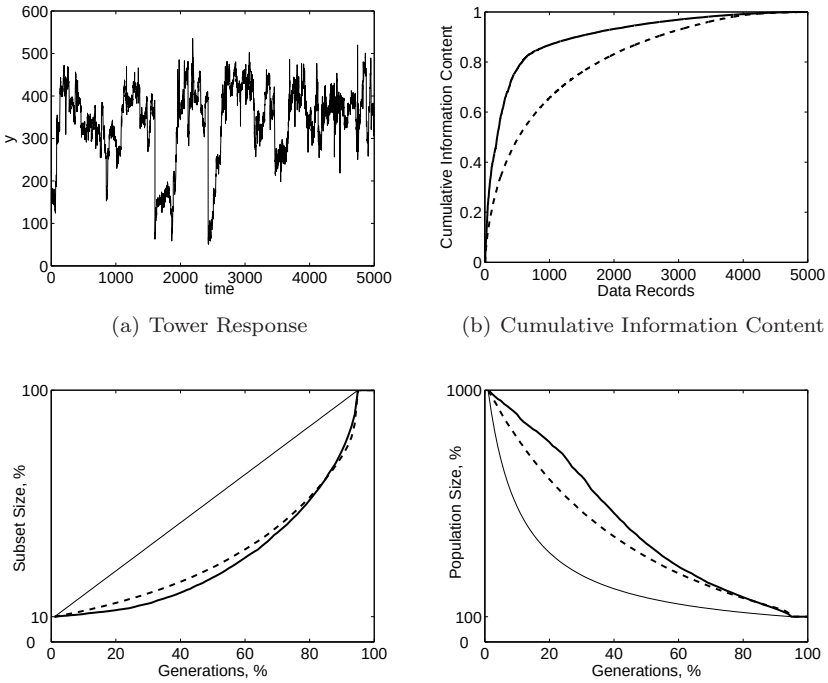
(a) Salustowicz2dBioN data

(b) Cumulative Information Content

(c) Subset size per generation

(d) Population Size per generation

Figure 7.5: Salustowicz2dBioN data. Plot (a) represents the input data of the Salustowicz2dBioN problem with 3931 points and the contour plot of the Salustowicz2d function. Plot (b) represents the Cumulative Information Contents of the two rankings of the the training data obtained with the SMITS procedure with non-linearity weights (thick black line) and surrounding weights (dashed line). Plot (c) illustrates the scheme of addition of new points in the ESSENCE algorithm, calculated based on the CICs of the plot (b) and on the intention to start modeling with subsets of 10% of the original size and to increase the subset size over the first 95% of generations.



(a) Tower Response

(b) Cumulative Information Content

(c) Subset Size per Generation

(d) Population Size per Generation

Figure 7.6: Tower data. Plot (a) represents 5000 response samples of the Tower problem. Plot (b) represents the Cumulative Information Contents of the two rankings of the the Tower data obtained with the SMITS procedure in the subspace of five significant inputs with non-linearity weights (thick black line) and surrounding weights (dashed line). Plot (c) illustrates the scheme of addition of new points in the ESSENCE algorithm, calculated based on the CICs of the plot (b) and on the intention to start modeling with subsets of 10% of the original size and to increase the subset size over the first 95% of generations. Due to the necessity to perform screening runs of 20 generations to identify driving variables in the Tower problem, the schemes for the ESSENCE experiments are computed for 180 generations, while the schemes for Reference runs and Random runs are computed for 200 generations.

All test problems are modeled with symbolic regression via Pareto genetic programming. In all experiments of this chapter we used the tree-based Pareto GP system that exploits the minimization of the two-dimensional fitness function, consisting of numerical error and expressional complexity of GP individuals (see Chapter 3 for more details). At each generation we preserve an archive of a fixed number of elite solutions located *closest* to the Pareto front in the objective space of model complexity and model error. Crossovers between archive and population individuals and point mutations of the archive individuals are used as genetic operators. In the case of crossover, crossover points are selected from the same or similar levels of the parent trees. Constants are randomly generated from the interval $[-10, 10]$. Other settings are given in Table 7.1.

We use the percentage of the area under the Pareto front of archive solutions at the last generation as the primary measure for comparisons of the quality of final solutions;see Chapter 3. To compare the predictive capabilities of the solutions in extrapolation, we also use the root of the mean squared error of the archive solutions on the test data [3].

7.3.3 Empirical results

To exclude the outlier GP runs from the comparison, we selected the top 75% of the runs for each regression problem. The box plots in Figures 7.7(a), 7.8(a), and 7.9(a) represent the percentages of areas under Pareto fronts at the last generation. The box plots in Figures 7.7(b), 7.8(b), and 7.9(b) represent the comparison of the root mean squared errors of final solutions on test sets requiring extrapolation. We emphasize the experiments for the primary comparison in bold: Reference, Random_Linear, EssenceSurrounding and EssenceNon-Linearity.

For all three problems we see that the ESSENCE experiments significantly outperform not only the reference runs but also the soft ordinal runs on random subsets (Random_Linear) for both performance measures. We use the Wilcoxon-Mann-Whitney (ranksum) tests for comparing the medians of Pareto front area percentage and RMSE over 75 independent runs. The p-values of these tests are given in Figure 7.2.

As we expected, the ESSENCE runs based on the ranking of data with respect

[3]The errors of the solutions of the **Tower** problem are compared on the training set.

Table 7.1: GP Parameters used in Empirical Experiments (unless stated otherwise).

Number of independent runs	100
Budget per generation	$100N$ function evaluations (N is the number of records)
Total number of generations for Reference runs	220
Total number of generations for Essence runs of Tower problem	180
Total number of generations for all other runs	200
Reference Population size	100
Archive size	100
Population tournament size	7
Archive tournament size	5
Crossover rate	0.95
Mutation rate	0.05
Rate of mutation on terminals	0.3
Basic Function Set	$+$, $-$, $*$, $/$, $square$, x^{real}, $x + real$, $x \cdot real$, e^x, e^{-x}
Function Set for Salustowicz2dBioN	Basic Set, $\sin x$, $\cos x$
Fitness Measure	Normalized Sum of Squared Errors
Complexity Measure	Expressional Complexity
Performance Measure 1	Percentage of Area Under Pareto Front
Performace Measure 2	Root of the Mean Squared Error on the TEST data

to the surrounding weight functional show a slightly better behavior, which is statistically better for the Tower problem for both performance measures, and is statistically the same for the KotanchekImbalanced and Salustowicz2dBioN problems.

For the Salustowicz2dBioN and Tower problems the ESSENCE experiments are statistically outperforming all other experiments with the exception of the last experiment of incremental addition of points from a Salustowicz2dBioN data, permuted randomly, and added to the training subset at the 'speed' equal to the 'speed' of the EssenceNonLinearity experiment. The results of

this NestedRandomOrderNonLinearity experiment are statistically the same for the Salustowicz2dBioN data, however are significantly worse for the two other problems. We explain the good performance of the experiment corresponding to NestedRandomOrderNonLinearity by the nature of the Salustowicz2dBioN data, which is heavily over-sampled in the training region. Subsets of 10% of original size already contain 393 points, which—even when sampled randomly from the training set— may contain a sufficient amount of information about the Salustowicz2D response surface.

There is little difference between Random_Surrounding, NestedSurrounding_ Linear, and NestedNonLinearity_Linear for KotanchekImbalanced and Tower problems. For the Salustowicz2dBioN data the Random_Surrounding experiment is significantly under-performing compared with NestedSurrounding_ Linear and NestedNonLinearity_Linear experiments.

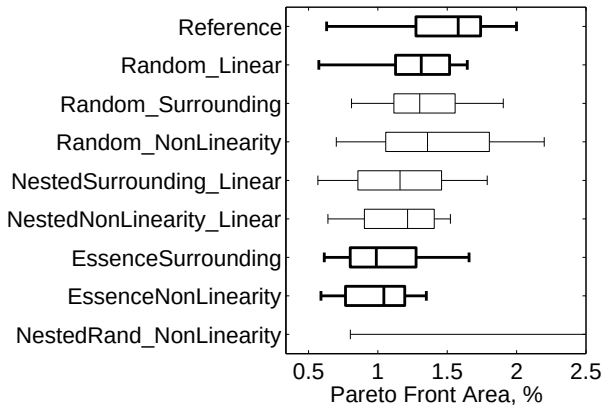
7.4 Summary

This chapter studies the performance of symbolic regression performed by means of incremental modeling of training data ranked in a special way. Our approach applies maximal modeling effort (measures by the size of the population) to subsets of 'essential' data records, and gradually increases the training subset size by adding new records, in the spirit of ordinal optimization. The new algorithm is called ESSENCE, and is expected to perform Effective Search Space Exploration by Nested Content-based Evolutions.

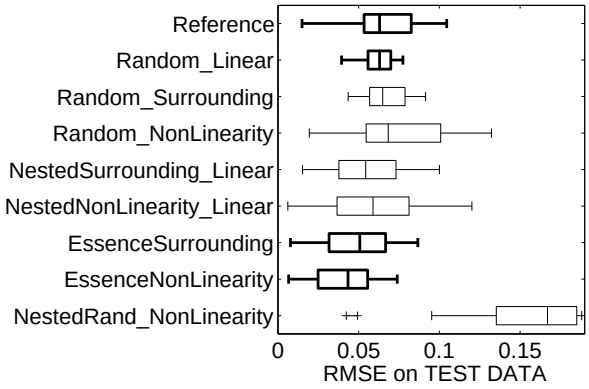
The special way of ranking the training data consists of (i) weighing the data with one of the weighing functionals, introduced in Chapter 2, and (ii) applying the iterative SMITS procedure to rank the weighted set in the order of decreasing importance, while inferring the cumulative information content of the ranked set.

An input-output data set ranked with the SMITS procedure has the property of nested balancedness, i.e. its first m records ($k < m < N$) have the maximal cumulative information content over other subsets of size m.

Evaluating the fitness of GP models on smaller subsets of the data, we can effectively increase the population sizes while keeping the number of function evaluations constant per generation. In soft ordinal Pareto GP, modeling is performed on random subsets of data with the size increasing linearly over the GP run, starting usually from 10% (see previous chapter). The linear increase

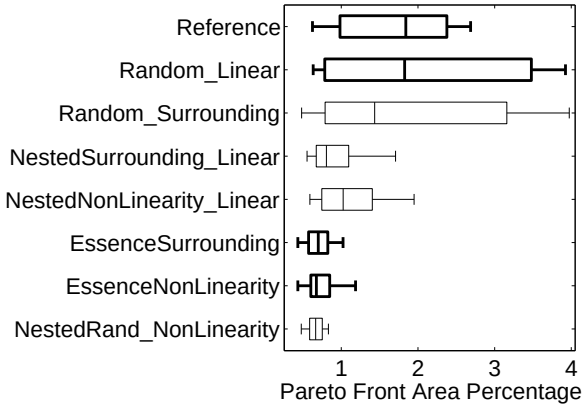


(a)

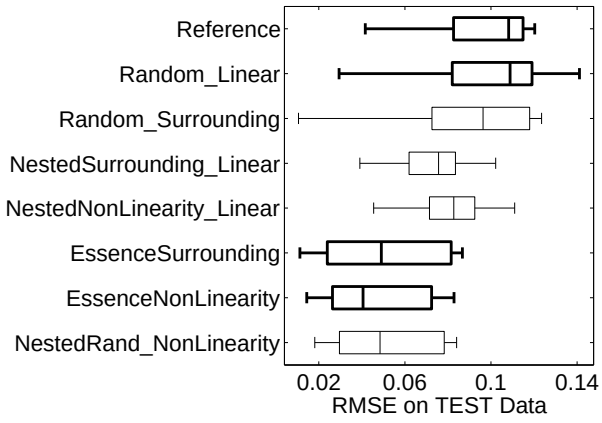


(b)

Figure 7.7: Empirical Results on the KotanchekImbalanced data. Box-plots corresponding to the primary comparison experiments are emphasized in bold.

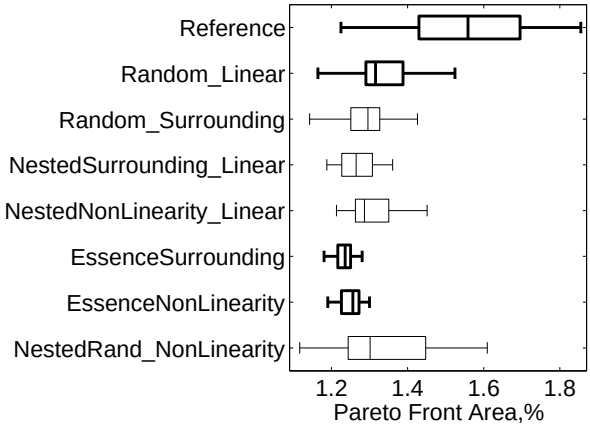


(a)

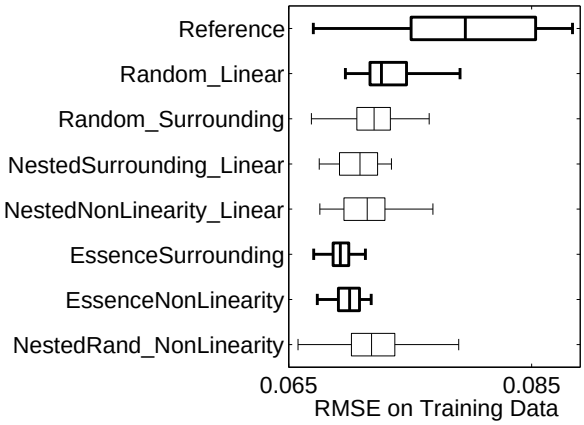


(b)

Figure 7.8: Empirical Results on the Salustowicz2dBioN data. Box-plots corresponding to the primary comparison experiments are emphasized in bold. Interestingly enough, the results of the NestedRand_NonLinearity experiments corresponding to an incremental modeling of the data records, randomly permuted before the modeling, produces results statistically indifferent from the results of the ESSENCE experiments. This may be related to the over-sampled nature of the Salustowicz2dBioN problem and to an accidentally good initial random permutation. Further analysis of the Salustowicz2dBioN problem confirmed that other random permutations of the data set may cause deterioration of the results.



(a)



(b)

Figure 7.9: Empirical Results on the Tower data. Box-plots corresponding to the primary comparison experiments are emphasized in bold.

Table 7.2: Part I. Significance of the comparisons of the experiments on KotanchekImbalanced and Salustowicz2dBioN test problems. The last eight columns of the Table represent the p-values of the Wilcoxon-Mann-Whitney test of comparing the medians of the distributions of the percentages of areas under Pareto Fronts and of RMSE errors of solutions over 100 independent GP runs. Smaller values of sample medians of both measures are preferred. All test are performed at the 95% confidence interval. Primary comparison groups are emphasized in bold.

Problem	Median	REFERENCE	Random_Linear	Random_Surrounding	NestedSurrLinear	NestedNonLinLinear	EssenceSurrounding	EssenceNonLinearity	NestedRandOrderNonLin
KotanchekImbalanced									
Pareto Front Area, %									
Reference	1.5792	–	–	–	–	–	–	–	0.
Random_Linear	1.3129	0.01	–	–	–	–	–	–	0.
Random_Surrounding	1.3016	0.04	0.73	–	–	–	–	–	0.
NestedSurrounding_Linear	1.1594	0.00	0.10	0.04	–	0.91	–	–	0.
NestedSurrNonLinearity	1.2143	0.	0.10	0.04	–	–	–	–	0.
EssenceSurrounding	0.9882	0.	0.	0.	0.25	0.16	–	0.61	0.
EssenceNonLinearity	1.0428	0.	0.	0.	0.10	0.02	–	–	0.
NestedRand_Nonlinearity	3.6953	–	–	–	–	–	–	–	–
RMSE on Test Data									
Reference	0.0630	–	–	0.95	–	–	–	–	0.
Random_Linear	0.0630	0.18	–	0.12	–	–	–	–	0.
Random_Surrounding	0.0649	–	–	–	–	–	–	–	0.
NestedSurrounding_Linear	0.0543	0.04	0.30	0.01	–	0.77	–	–	0.
NestedNonLinearity_Linear	0.0589	0.15	0.63	0.09	–	–	–	–	0.
EssenceSurrounding	0.0507	0.01	0.04	0.00	0.36	0.36	–	–	0.
EssenceNonLinearity	0.0434	0.	0.	0.	0.02	0.04	0.17	–	0.
NestedRand_Nonlinearity	0.1670	–	–	–	–	–	–	–	–
Salustowicz2dBioN									
Pareto Front Area, %									
Reference	1.840	–	–	–	–	–	–	–	–
Random_Linear	1.820	0.37	–	–	–	–	–	–	–
Random_Surrounding	1.435	0.83	0.58	–	–	–	–	–	–
NestedSurrounding_Linear	0.806	0.	0.	0.00	–	0.07	–	–	–
NestedNonLinearity_Linear	1.024	0.	0.00	0.04	–	–	–	–	–
EssenceSurrounding	0.700	0.	0.	0.	0.00	0.	–	–	–
EssenceNonLinearity	0.675	0.	0.	0.	0.01	0.	0.8	–	–
NestedRand_Nonlinearity	0.665	0.	0.	0.	0.	0.	0.29	0.45	–
RMSE on Test Data									
Reference	0.1082	–	–	–	–	–	–	–	–
Random_Linear	0.1098	0.54	–	–	–	–	–	–	–
Random_Surrounding	0.0963	0.77	0.42	–	–	–	–	–	–
NestedSurrounding_Linear	0.0756	0.	0.	0.00	–	0.09	–	–	–
NestedNonLinearity_Linear	0.0827	0.	0.00	0.01	–	–	–	–	–
EssenceSurrounding	0.0491	0.	0.	0.	0.00	0.	–	–	–
EssenceNonLinearity	0.0405	0.	0.	0.	0.	0.	0.44	–	0.26
NestedRand_Nonlinearity	0.0485	0.	0.	0.	0.00	0.	0.84	–	–

Table 7.2: Part II. Significance of the comparisons of the experiments on Tower problem. The last eight columns of the Table represent the p-values of the Wilcoxon-Mann-Whitney test of comparing the medians of the distributions of the percentages of areas under Pareto Fronts and of RMSE errors of solutions over 100 independent GP runs. Smaller values of sample medians of both measures are preferred. All test are performed at the 95% confidence interval. Primary comparison groups are emphasized in bold.

Problem	Median	REFERENCE	Random_Linear	Random_Surrounding	NestedSurrLinear	NestedNonLinLinear	EssenceNonLinearity	NestedRandOrderNonLin
Tower Problem								
Pareto Front Area, %								
Reference	1.5587	–	–	–	–	–	–	–
Random_Linear	1.3160	0.	–	–	–	–	–	–
Random_Surrounding	1.2958	0.	0.0220	–	–	–	–	0.3800
NestedSurrounding_Linear	1.2648	0.	0.	0.0934	–	0.01	–	0.0311
NestedNonLinearity_Linear	1.2868	0.	0.0303	0.6968	–	–	–	0.7592
EssenceSurrounding	1.2362	0.	0.	0.0000	0.0079	0.	0.01	0.0001
EssenceNonLinearity	1.2563	0.	0.	0.0007	0.1995	0.	–	0.0066
NestedRand_Nonlinearity	1.3015	0.	0.3579	–	–	–	–	–
RMSE on Training Data								
Reference	0.0795	–	–	–	–	–	–	–
Random_Linear	0.0726	0.	–	–	–	–	–	–
Random_Surrounding	0.0720	0.	0.0527	–	–	–	–	–
NestedSurrounding_Linear	0.0709	0.	0.0000	0.0238	–	0.09	–	0.0715
NestedNonLinearity_Linear	0.0715	0.	0.0098	0.5028	–	–	–	0.6217
EssenceSurrounding	0.0693	0.	0.0000	0.0000	0.0008	0.	0.03	0.0000
EssenceNonLinearity	0.0700	0.	0.0000	0.0000	0.0444	0.	–	0.0007
NestedRand_Nonlinearity	0.0718	0.	0.0914	0.9049	–	–	–	–

of the subset size, however, may not be the best way to improve the fidelity of fitness evaluations in the course of a run.

This chapter has introduced a new scheme for increasing the size of training subsets that depends of the training data, and can be used either with random subsets (Soft Ordinal Pareto GP), or with nested subsets of the ranked data (the ESSENCE algorithm). In the new subsetting scheme the subset size grows slower than linearly, and hence, allows the use of larger population sizes within the same computational budget.

The experiments of this chapter are designed to compare the effects of using

random and nested subsets with sizes growing linearly and non-linearly over the GP run. Three regression problems, all containing imbalanced data, are used for these experiments. Results of the experiments indicate that the new incremental modeling approach on nested subsets of the training set ranked with the SMITS procedure, with non-linear increase in the subset size, produces results that are significantly than results of the reference Pareto GP runs and of the soft ordinal Pareto GP runs in the old setting (random subsets increasing linearly). The comparison is based on the values of two performance measures: percentage of area under the convex hull of the archive solutions in the objective space of model expressional complexity and model accuracy for the training data, and the best root mean squared error of archive solutions on the test data.

A statistically significant difference between effects of the surrounding and non-linearity weights is present only on the Tower problem, where the ESSENCE algorithm applied to data ranked with surrounding is outperforming the ESSENCE algorithm applied to non-linearity based ranking. This supports our recommendations in Chapter 5 to use the surrounding weights instead of the non-linearity weight for 'very imbalanced' problems, where the average distance to k nearest-in-the-input-space neighbors varies significantly among the training points.

The ESSENCE algorithm seems to be the best way to produce considerably better solutions with the same computational budget. We speculate that the incremental modeling by the ESSENCE algorithm may also produce 'better solutions faster', since the training data can be sensibly compressed to a smaller subset of similar information content, and incremental modeling can be stopped much earlier than the standard regression runs enable. An important property of the ESSENCE algorithm is the possibility to get a reliable set of intermediate solutions from the GP system, almost at all times. Due to the fact that even at the beginning of the run the problem is modeled on a subset of data containing the highest information content with high population sizes, *good* solutions can be discovered in the early stages (if information content of the starting training subset is sufficiently high), and are being tested further on prototype data in the course of a run.

In further research we wish to incorporate automatic variable selection, automatic dimensionality reduction, and automatic selection of important records into the ESSENCE algorithm, and to develop early stopping criteria for incremental

evolutions presented in this chapter. Intelligent evaluation of competing individuals in a spirit of racing would be desired as well.

Part III

Problem Analysis and Reduction

8

Problem Analysis and Reduction

This chapter summarizes some of the current methods for model interpretation, problem analysis and reduction employed by Pareto genetic programming. Practical considerations of convergence identification, variable selection, and ensemble selection are presented.

8.1 Analysis of problem difficulty

According to the framework of iterative model-based problem solving, the stage of problem analysis and reduction follows the stage of model development. As we said in Chapter 3, the stage of model development in symbolic regression itself is an iterative loop over three phases of model generation, evaluation, and selection (see Figure 8.1. In principle, several iterations of evolutionary search as well as several replications of evolutions are required at the stage of model development, to produce individuals of sufficient quality that can be interpreted at the problem analysis and reduction stage. The question on how many iterations and how many replications to do is an open research question. The answer obviously depends on the available time budget and on the knowledge on reproducibility of the evolutionary strategy. While for research papers we perform 50 to 100 replications to obtain sufficiently large samples for comparing statistical significance of performance improvements of particular strategies, in practice only few replications are sufficient for interpreting the results and drawing preliminary conclusions.

The question on the necessary number of generations is a tricky one, due to the fact that initially neither anything is known about the problem difficulty, nor a realistic level of prediction accuracy can be estimated. The time required to obtain solutions of sufficient quality is hard to estimate. For this reason the initial cycles over data-model-problem analysis are short due to the short model development stage.

Even if there are no strict time constraints, it may not be very practical to perform the evolutionary search or the fixed values of internal parameters for too long, since it is known to stagnate at the large number of iterations. The problem of stagnation, or loss of performance improvement is directly related to the balance of exploration and exploitation in the evolutionary search. If exploration of the new areas of the search space of alternative solutions deteriorates over the evolutionary run, then excessive exploitation of discovered solutions may lead to convergence, or inbreeding, when the population of individuals converges to the same solution due to diversity loss. In archive-based Pareto GP as opposed to the standard GP the problem of inbreeding is effectively tackled by re-initialization of the population of individuals at regular intervals. The existence of the archive with Pareto-aware control of prediction accuracy and model complexity guarantees diversity of the archive. Avoidance of convergence of the population and the archive in Pareto GP, however, does not eliminate the problem of performance stagnation at the high number of generations. If we model the data for too long, at some point we will observe stagnation, and it is practical to stop the evolutionary process as early as possible, after stagnation is achieved.

The measure according to which the performance progress of a symbolic regression run is evaluated and monitored, makes a lot of difference in determining stagnation period. Standard GP most often exploits the fitness landscapes - the graphs of the best fitness in the population per generation. In Pareto GP this performance measure corresponds to the best fitness in the archive per generation. Since the archive is updated using non-dominated sorting, the procedure guarantees that the best error in the archive does not get worse during the run. This implies that the landscapes of the best fitness in the archive are monotonously increasing.

Due to the fact that the measure for the best fitness in the archive does not fully capture the overall quality of the archive solutions, different performance

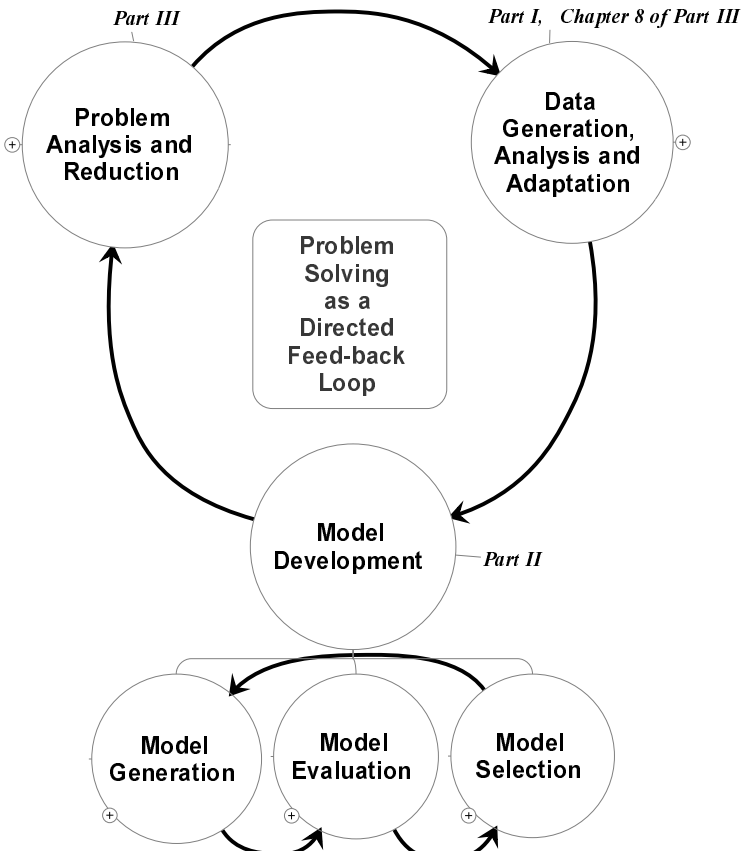


Figure 8.1: Generic scheme for model-based problem solving as a loop over data generation, analysis and adaptation, model development, and problem analysis and reduction.

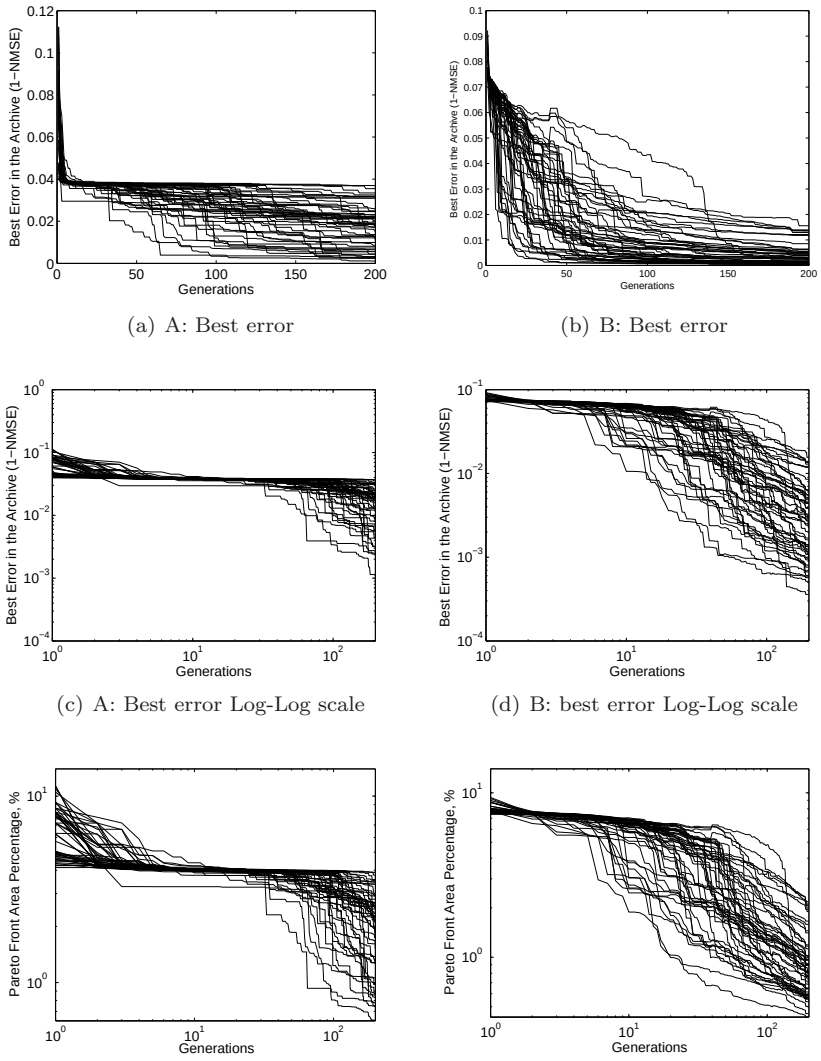
measure is often used to monitor the quality of symbolic regression - the percentage of the area under Pareto front in the model complexity vs. model error objective space (see Chapter 3).

One practical drawback of this theoretically founded measure is the additional computation time required for its evaluation. In cases where time is crucial and all computations must be reduced to the essence, we advise to use the "best error" measure for on-line monitoring of the evolutionary progress, and use the "area percentage" for off-line analysis of obtained solutions, and *a posteriori* analysis of problem difficulty. In principle, the user may decide to visually monitor the performance of the evolution by observing snapshots of the archive plotted in the objective space of model complexity and model error. With well presented graphics (like for example the one produced by the ParetoFrontLogPlot function in the DataModeler add-in for Mathematica) it is easy to observe whether the Pareto front of the archive is being pushed towards zero during the evolution, and exact computations of the Pareto front area may be omitted for speed.

Irrespectively of which measure is selected for performance monitoring, it is convenient to plot it on the log-log scale over the generations. A rule of thumb in deciding that evolution is *progressing* is the observation that the graph of its performance measure corresponds to a line on a log-log scale.

In Figure 8.2 we give examples of two experiments performed on the (hard) Salustowicz2dBio data from Chapter 7. Experiment A corresponds to the standard Pareto GP runs (plots at the left-hand side of Figure 8.2). Experiment B corresponds to EssenceSurrounding runs from Chapter 7(plots at the right-hand side). Both experiments are performed with the same budget of function evaluations. Each experiment consists of 50 independent replications over 200 generations. Plots 8.2(a) and 8.2(b) present the graphs of the best error in the archive plotted in the linear scale[1]. Note that examination of the performance of experiment A may entice to draw a conclusion on stagnation of experiment A. When the same graphs are plotted on the log-log scale (see Figure 8.2(c) and 8.2(d)), the steady performance increase is clearly observed in both experiments. The slope of the line (approximating the performance measure or the mean of the performance measure in the log-log scale) may be interpreted as the speed of convergence to solutions, or as the speed of the evolutionary progress. Clearly,

[1]Note that for the ESSENCE runs monotonicity of the fitness landscape in not guaranteed, since the training size is changing at each generation according to the ESSENCE algorithm.



(a) A: Best error

(b) B: Best error

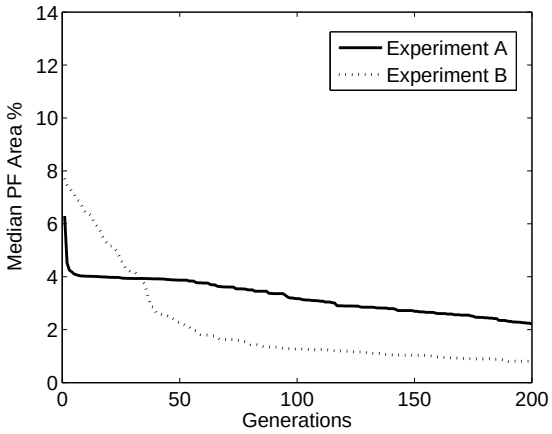
(c) A: Best error Log-Log scale

(d) B: best error Log-Log scale

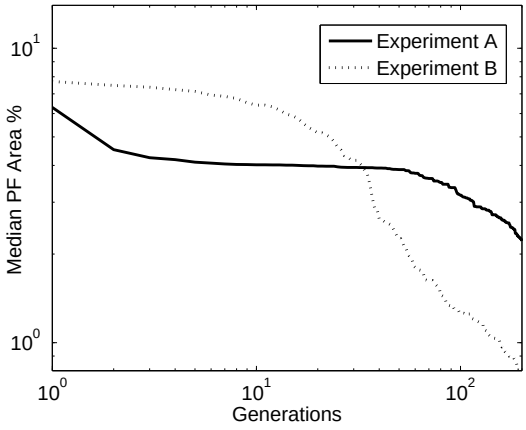
(e) A: PF Area % Log-Log scale

(f) B: PF Area % Log-Log scale

Figure 8.2: Monitoring performance measure on log-log scale is informative indicator of evolution performance, and the potential to achieve performance improvement. By observing plots (c)-(e) we can certainly conclude that the system would have achieved a performance improvement in the experiment A as well as in the experiment B, if runs would have continued for a larger number of generations.



(a) Median best error, liner scale



(b) Median best error, log-log scale

Figure 8.3: The slope of the value of the performance measure per generation plotted in the log-log scale can be interpreted as the speed of performance improvement, or the speed of evolution. The increment of this median in the case of Pareto Front area percentage can be interpreted as a difficulty or a progress of the Pareto GP system in modeling data at certain generation. As long as the median performance measure is close to a line in the log-log scale - evolution progresses.

experiment B achieved a higher speed of convergence to solutions than experiment A in both performance measures (see Figure 8.3).

The speed of evolutionary progress can also be used as an indicator of problem difficulty at a particular generation. For the experiments in the example above, by observing the speed of progress on the Figures 8.2(e) and 8.2(f) we conclude that experiment B does not seem to be difficult for the Pareto GP system (which implements the ESSENCE algorithm in this case), while experiment A is rather difficult, since the speed of the performance improvement achieved by generation 200 is relatively slow. To conclude this example, we add that it would be incorrect to state that in Figure 8.3 the Pareto GP system does not have a potential to come up with an archive of sufficiently high quality, but it would be plausible to hypothesize that achieving a sufficiently high quality of solutions (and hence, a sufficiently low Pareto front area percentage) may take from 10^3 to 10^4 generations, if stagnation does not happen.

8.2 Analysis of problem dimensionality

Spurious variables are one of the major culprits of problems with industrial data analysis and empirical modeling, and especially with design and analysis of experiments. Reducing the data space to a subspace of significant variables is crucial for developing robust and simple regression models. In Table 1.1 (on the high-level comparison of some of the regression methods) we indicated that all non-evolutionary methods have a high risk of producing solutions that contain insignificant inputs. This danger may (most certainly) imply a fast deterioration of performance of the final solutions when more irrelevant variables are added to the data.

Of course, any researcher practicing empirical modeling (including the author) will assure that the application of a modeling method should be preceded by a scrupulous data analysis and pre-processing step. The question arises how to perform this data pre-processing step to exclude insignificant variables from the data. Strange as it might seem, not many researchers practicing empirical modeling with *parametric* methods will understand this question at all. With respect to the goals of parametric methods, irrelevant variables are not a threat - since the primary task is is to generate an optimal solution (of a minimal order) that minimizes the prediction error out-of-sample. This goal does not imply that insignificant variables should be excluded. Especially for the training accuracy, adding a few irrelevant variables may actually improve the prediction

error of produced models - we refer to this phenomenon as to over-fitting. Examples of such a situation can often be observed in symbolic regression: models with a higher than needed number of variables obtain a slight improvement in prediction error at the expense of increased complexity. Unlike parametric methods optimizing the error, symbolic regression pursues a slightly different (and loosely formulated) goal - to identify the *optimal space* and *optimal model structures* that convincingly describe the observed response as a function of the identified driving inputs.

Those practitioners who worry about the presence of insignificant inputs do not have too many options to identify them beforehand. The best but also the most time- consuming method models all subsets of input variables, and analyzes the results with respect to prediction error and robustness. For dimensionality 10-1000 this is a hardly practical approach. Another approach would be to perform a principal component analysis or a factor analysis to reduce the problem dimensionality to a smaller number of meta-variables which are linear combinations of the original variables, or to extract the latent dimensionality of the problem and get some knowledge about the number of factors containing the same information. The potential problem of these 'classical' approaches used for analysis of data generated by a non-linear system, is the fact that they only take into account mutual linear correlations between variables, and hence have limited capabilities in analyzing the relevance of non-linear combinations of inputs to the output. Besides, they do not select but create new variables in the new reduced set, eliminate multicollinearity (which is most often present in real measurements), and are sensitive to outliers, which heavily reduces applicability of these approaches to analysis and plausible interpretation of (messy) industrial data.

There are several variable selection strategies developed in the field of machine learning (mostly for classification problems). They can be divided into two groups - filter strategies and wrapper strategies. In a filter approach a subset of variables is selected according to a certain criterion, and then is supplied to an induction engine for model development. In a wrapper approach, variable selection is guided by the induction engine (i.e. is wrapped around the engine), and depends on the performance of models.

Variable selection through wrapping is introduced in (Kohavi and John, 1997) with the goal of selecting a subset of optimal variables that minimizes

the prediction error of a classifier. The paper is interesting for many reasons, particularly because the authors explore the relation of optimal variables to their relevance, give formal definitions of variable relevance (albeit for classification), and analyze the shortcomings of filter algorithms. The relevance is defined in terms of the Bayes classifier - the optimal classifier for a given problems. The variable is called strongly relevant, if its removal causes a performance deterioration of the Bayes classifier. The variable x is called weakly relevant, if it is not strongly relevant, and there exist a subset of variables S, such that the performance of the Bayes classifier on S is worse than its performance on $S \bigcup x$. The authors also emphasize that finding a subset of optimal variables is not the same as finding a subset of relevant variables, if optimality is understood as minimization of prediction error. This agrees with our observation of symbolic regression which will improve the prediction error by using additional irrelevant variables, if not discouraged to do so.

Kohavi and John (1997) suggest to directly apply a wrapper approach to forward selection or backward elimination of optimal variables. They prefer a computationally cheaper heuristic for incremental forward selection, which creates nested subsets of optimal variables starting from an empty subset. A simplistic explanation would looks as follows. At the first step, the model induction engine creates d models using single variables $x_1,\ldots,x_d$, and selects the 'optimal' variable that implies a model with the minimal prediction error. At the second step, the engine produces $d-1$ models on two variables, one of which is the optimal variable selected at the first step. Two variables are selected and used in a combination with the remaining $d-2$ variables for the third step of the wrapper approach with forward selection, and so on. In backward elimination, the first step produces d models containing all combinations of $d-1$ variables, and then variables are iteratively eliminated from the selected optimal subsets. Obviously, the approach is heuristical in nature, since only particular subsets of variables are considered, but heuristics and meta-heuristics are the current practical reality of most variable selection methods due to the size of data being considered.

An important general feature of wrapper methods is the fact that they look for subsets of optimal rather than relevant variables, and optimality depends on biases and accuracy estimation of a particular induction engine.

One of the unique capabilities of genetic programming is a built-in power

to select significant variables for constructing models and to gradually omit the variables that are not relevant for describing the response. Variable selection based on genetic programming has been exploited in various applications of industrial data analysis, where the significant inputs are generally unknown, or their number needs to be minimized (for examples see (Francone et al., 2004; Gilbert et al., 1998; Landry et al., 2006; Neshatian et al., 2007; Poli, 1996; Sherrah et al., 1997; Yu et al., 2007)).

We speculate here, that genetic programming incorporates the features of the wrapper approach for selection of optimal variables blended directly into the evolutionary search - relevant variables that are crucial for describing the response must be present in high-fitness individuals, and variables, present in high fitness individuals are candidates for being optimal variables. A complication is that optimality depends on a particular stage of evolution, since high fitness is a relative notion in evolutionary search. Anyway, the usual approach to variable selection based on GP is to analyze variable presence in the best equations, and infer variable sensitivity based on the presence rates (and average it over independent runs, or over independent experiments with different GP settings).

In (Smits et al., 2005) a variable selection strategy based on Pareto GP was introduced. In the remainder of this section we shall highlight its main features and speculate on the reasons for its good performance.

Variable sensitivity analysis is based on the assumption that fitness of a variable is related to fitness of the host individual. On one hand, high-fit individuals should contain high-fit variables (compare with optimality feature given above). On the other hand, not all variables may be equally important in a given individual, and variables present in a high-fitness individual are not necessarily relevant (in particular, variables participating in inactive code are irrelevant, e.g. $... + x_1/x_1 - x_2/x_2$). This raises the question on how to assign credits to variables to obtain plausible sensitivities.

Smits et al. (2005) introduced a procedure for variable sensitivity analysis in Pareto GP through a straightforward fitness inheritance scheme of *a posteriori* credit assignment. The main idea of this scheme is to uniformly distribute the fitness of the host individual over all variables present in the individual, and then sum up the obtained scores over all occurrences of a variable in a selected set of individuals and use the obtained cumulative scores as sensitivities of the selected set of variables to the performance of the selected set of individuals.

Such credit assignment based on fitness inheritance, can be applied either to the total population (or to any subset of individuals from a population), or to the archive, or to the Pareto front individuals from the archive[2].

A posteriori sensitivity inference based on the straightforward fitness inheritance applied to the Pareto front individuals of the archive seems to be a more robust way to avoid sensitivity distortion caused by unimportant variables that might be present in the archive equations. Since the Pareto front is a set of optimal (non-dominated) trade-offs in the space of model complexity and model error, the Pareto front solutions of the minimal model error will by definition have a minimal expressional complexity, possible for the given error. This suggests from all high-fitness individuals, those of minimal complexity will be chosen, which are likely to have less inactive code. Those variables that are present in high-fit individuals but are irrelevant for describing the response, will likely be present in low-fitness individuals in smaller fractions than relevant variables. Therefore, the overall cumulative score of such irrelevant variables is likely to be smaller, than the overall score of relevant variables, and the sensitivity analysis based on the Pareto-optimal archive solutions will likely identify relevant variables rather than optimal variables only. To further improve the differentiation between relevant variables, which are crucial for describing the response, and optimal variables, which are present in the high-fitness equations, we suggest to perform sensitivity analysis using fitness inheritance on the subset of archive solutions located at the knee of the Pareto front (in the low error-low complexity region).

We speculate, that the appropriateness of the variable sensitivity computed through the fitness inheritance scheme directly depends on the model selection strategy, used for identifying the set of individuals to which fitness inheritance should be applied. For example, even if models are developed with the standard GP method and single objective fitness measure, it seems appropriate to perform the sensitivity analysis not on all population individuals but on the Pareto front individuals in the complexity/error space. In this case, the additional computational effort required for complexity evaluation of population individuals is minimal, since for standard GP the calculations should be performed only once after the solutions are obtained through the standard GP run. The Pareto-based model selection will drive variable selection towards discovery of relevant (in

[2]The the number of individuals in the archive is fixed, and may contain a lot more individuals than there are on the Pareto front in a selected complexity versus model error objective space.

the sense of complexity/accuracy trade-off) rather than optimal (in a sense of accuracy) variables.

It is also important to emphasize that model selection strategy applied to all sub-expressions of individuals seems to be more robust than selection applied to particular individuals (expressions associated with the root-nodes for tree-based representation). This requires minor additional computation effort (for evaluating finesses of all intermediate nodes in the individual), but produces more reliable statistics. Higher reliability also comes from the fact that the impact of inactive code on the sensitivity is reduced, since variables that are present in inactive subexpressions are likely to get lower cumulative scores compared with variables present in active sub-expressions, which do influence the high fitness of the host individual.

In Pareto GP, variable sensitivity analysis of archive equations is executed through the entire run - this allows to implement adaptive variable selection on-line[3].

It is possible to imagine that more criteria than just expressional complexity and error can be incorporated into into credit assignment and model selection for variable sensitivity in order to have a positive impact on the identification of relevant variables. For example, niching of individuals per dimensionality is likely to achieve this goal, since variables that are present in low-dimensional high-fitness individuals are likely to be more relevant than variables present in high-dimensional high-fitness individuals.

In summary, is that variable selection and sensitivity analysis is an important step of model interpretation, is one of the most significant distinctive features of Pareto GP compared to other methods for variable selection and for model development, is performed with minor additional computations, and can *effectively* reduce dimensionality of the problem for the next Data analysis-Model development-Problem analysis cycles.

To conclude we emphasize that variable selection is performed automatically without a bias of the domain expert. Because of this fact, it is important to always confront the obtained sensitivities with the assessment of the problem owner. A critical assessment of variable importance will inevitably lead to increased insight into the problem. Two situations are possible - the domain expert either

[3]For the purpose of generality and fair comparison of methods, no adaptive variable selection was used in the experiments of this dissertation, since adaptive elimination of the least significant variables makes the search easy, especially at the initial stage.

accepts the sensitivity analysis, because it "makes sense"[4], or the domain expert does not accept it, since it does not make any sense at all. Both cases lead to new insight into the problem. If the variable selection is acceptable, it means that the dimensionality of the problem can be effectively reduced and a new problem solving cycle can be started. If variable selection is unacceptable, it is an important sign that the information which the expert is anticipating to retrieve is *not* present in the data.

8.3 Analysis of solution reliability

8.3.1 Model ensembles

A critical assessment of the reliability of solutions produced by the Model development phase is a major issue in model-based problem solving. Besides the numerous efforts done to incorporate the drive for generalization and interpretability into the model development phase itself, a close scrutiny of generated solutions is needed to extract reliable models appropriate for deployment.

The major challenge of using empirical models in industrial applications is the potential to produce dangerously wrong predictions in the unobserved regions of the input space. Since there is an infinite number of response surfaces perfectly fitting any given data set, we aim at discovering those that are the simplest or the smoothest, or have a predefined model structure (as in parametric methods). In general, if nothing is known about the behaviour of the underlying system (e.g. about the 'true' response surface), data-driven models can be fully trusted only at the data points (in noise-free cases). Whether we attempt to predict the response outside the observed region of inputs, or we predict the response within the observed region for new data records, in both cases model predictions should be treated with caution. The main challenge of model maintenance is to identify these moments, and alarm when predictions of reduced validity are about to be generated.

The problem of developing robust and *trustworthy* models with Pareto GP

[4]On hindsight, since unexpected or counter-intuitive variables can be selected as important ones, or some variables that are considered to be important turn out to have little relevance for a description of the response, the conclusion of the expert: "In principle it makes a lot of sense", often comes after some mind work, preceded by :"Hmmm, this is very strange - can't be true."

is addressed in recent studies by Kotanchek et al. (2007, 2008). The idea of assessing the trustworthiness of predictions is resting on the following generic tenet (see (Kotanchek et al., 2007)): If we can select an ensemble of models that are accurate and *diverse*, these models will be constrained to agree in predictions where there is data, and constrained to disagree in predictions where there is no data. The disagreement between these ensemble models, if determined as a function of the input space, can be interpreted as a measure of trustworthiness of the ensemble.

The idea of using ensembles for improved generalization of the response prediction is by far not new in regression, and has been extensively used in neural networks (see, for example,(Hansen and Salamon, 1990; Krogh and Vedelsby, 1995; Liu and Yao, 2002; Liu et al., 2000; Wolpert, 1992)), and even more extensively in boosting machine learning in general (albeit, mostly for classification), see (Folino et al., 2006; Freund, 1995; Freund et al., 1993; Iba, 1999; Paris et al., 2001; Schapire, 1990; Sun and Yao, 2006) for more examples.

Krogh and Vedelsby (1995) presented the idea of using the disagreement of ensemble models for quantifying the ambiguity of ensemble prediction for neural networks, but the approach has not proliferated in symbolic regression. Krogh and Vedelsby (1995) expressed the ensemble generalization error as the variance trade-off between the bias and the variance of the ensemble. They defined it as a difference between two quantities - the weighted sum of generalization errors on individual networks and the weighted average of ambiguities w.r.t. the selected ensemble, determined as the weighted sum of squared deviations of prediction errors at sampled points from the weighted mean of predictions. This last term of the weighted average of ambiguities is called ensemble ambiguity and contains all correlations of individual networks. The authors suggested to evaluate ensemble ambiguity on a random set of points sampled from the input space.

The approach of trustworthiness quantification introduced in (Kotanchek et al., 2007), attacks the notion of trust in symbolic regression via Pareto genetic programming (the approach can be seamlessly incorporated into standard GP as well). The ensembles are built from the archive individuals such that combinations of Pareto-optimal solutions, and solutions producing predictions with the smallest mutual correlations are selected. After the ensemble is built, its ambiguity (the term is called disconsensus) is defined as the prediction range at a selected point for small ensembles, or as the $0.1 - 0.9$ quantile

range. The prediction at a selected point $\mathbf{x}$ is computed as the median-average of ensemble predictions (the latter is defined as an average of three to five predictions surrounding the median prediction). The final ensemble prediction is an aggregate of predictions of all ensemble individuals, and the value of ensemble prediction at the arbitrary point $\mathbf{x}$ is defined as a pair of the median average prediction at $\mathbf{x}$, and the ensemble disconcensus at $\mathbf{x}$.

Note that unlike boosting methods aiming at improving the prediction accuracy through a combination of weak learners into an ensemble, symbolic regression aims at constructing model ensembles for estimation of the reliability of predictions of ensemble models.

The hardest part in building ensembles is to pick out a proper strategy to select individuals for an ensemble, such that the individuals provide similarly accurate predictions of the observed response at the given input points, but are as diverse as possible in all other aspects (the amount of publications on ensemble selection for machine learning, and especially for NNs appearing since the '90s is a confirmation of a great challenge).

A heuristical method presented in (Kotanchek et al., 2007) pursues the goal of combining multiple criteria in diversity definition - Pareto optimal solutions represent the ultimately diverse solutions with respect to individual's complexity (expressional, or non-linearity-based) and prediction error, while the uncorrelated models represent the solutions that are (sub-optimally) least correlated with respect to prediction error.

It is worth to note that if the diversity of the prediction error were the only important criterion for selecting models into an ensemble, the selection problem could have been elegantly and explicitly formulated as a quadratic minimization problem. If the task is to select e_B ensemble individuals from a set of B best individuals $\{\hat{f}_1, \ldots, \hat{f}_B\}$, accurately predicting the observed response such that they are least correlated with respect to the prediction error, then the solution can be identified as a solution of the following optimization problem:

$$\min \sum_{1 \leq i \leq j \leq B} c_{i,j} z_i z_j,$$

$$s.t. \sum_i z_i = e_B,$$

where z_i is the binary variable indicating whether individual $\hat{f}_i$ is selected into the ensemble, and $c_{i,j} = \left| R\left(\hat{f}_i(\mathbf{x}), \hat{f}_j(\mathbf{x})\right)\right|$, is the absolute value of the correlation coefficient between the responses predicted by $\hat{f}_i$ and $\hat{f}_j$. This problem, called a Quadratic Knapsack problem, has been studied extensively in operations research, and there are many exact and heuristic methods for solving it. The only warning that needs to be made is that if we do not want to build ensembles from over-fitting models, we need to explicitly include a minimization of generalization error into the problem formulation, and make it at least a bi-objective minimization problem. One of the options is to use a composite minimization function with an added term corresponding to the generalization error, but the general question on how to best incorporate multiple objectives for ensemble selection in the best way remains open.

8.3.2 Application for sequential designs

The primary goal of constructing model ensembles in Pareto GP is computation of the ensemble disagreement, which is used as a trust metric for ensemble prediction (this goal is more important than the reduction of the prediction error). Another benefit of constructing model ensembles lies in the application to adaptive data collection. Targeted data collection appeared as a side application of the ensemble trust metric, and seems to become one of the most significant applications of trustworthiness identification of data-driven models. When an unknown function has to be reliably re-constructed in a high-dimensional space, and collection of data (measured or simulated) is severely limited—due to intolerable time requirements, financial expenses or both—adaptive data collection and modeling is one of the few solutions for tackling this problem.

Model-based problem solving through adaptive data collection is a natural approach to learning, where the knowledge of the unknown underlying system is gathered incrementally through a series of steps of data collection, model development, and problem interpretation. It therefore naturally fits to a generic scheme depicted in Figure 8.1, where an arbitrary modeling method is used for Model Development rather than the depicted evolutionary search.

Incremental learning through the iterative process of experimental design, empirical model building, and analysis of the model is classic in the response surface methodology approach (RSM) to optimization; see (Box and Draper,

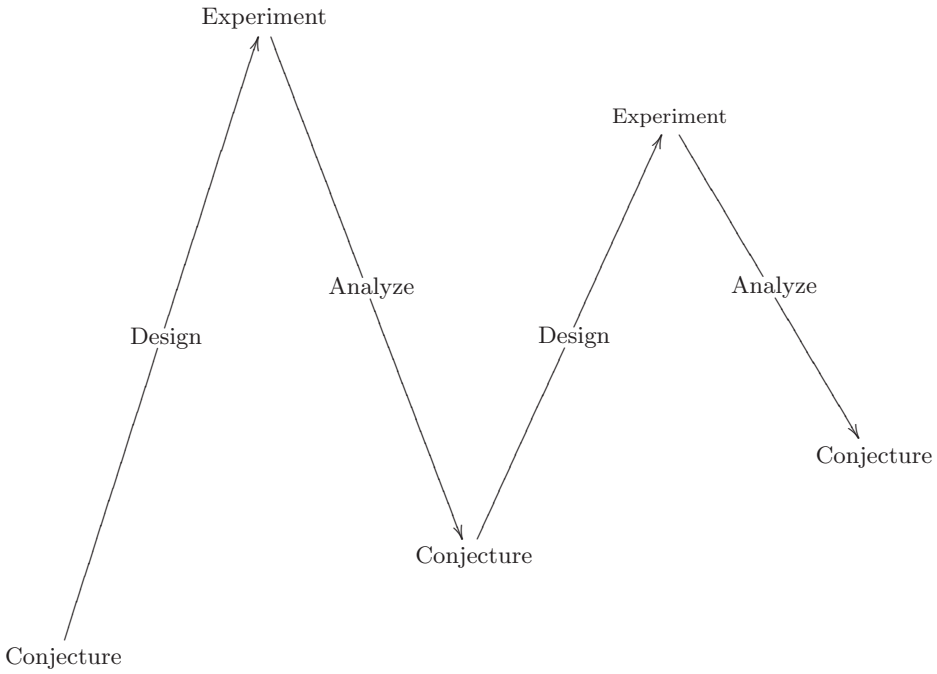


Figure 8.4: Response surface methodology as an iterative process of knowledge collection through validating or denying conjectures made and re-fined iteratively.

1986; Khuri and Cornell, 1987; Kleijnen, 2008b). RSM is used for optimization of an unknown system, but also for mapping the system into the response surface. In a general sense, RSM can be seen as an approach of understanding the given simulation system. In such 'learning' by means of RSM the user has to repeatedly perform the conjecture-design-experiment-analysis steps to come to an appropriate model of the response surface that can be adequately optimized, see Figure 8.4 adopted from (Box and Draper, 1986).

In RSM the user has to repeatedly (1) make a conjecture on the type of the parametric model (appropriate model structure), and the area of the input space (design space) where the data will be collected, (2) design the experiments in the selected input region (*for the selected model type*) such, that the number of calls to a system is minimal, (3) perform the experiments (note that until all experiments are finished, nothing is supposed to be done to incrementally interpret preliminary results of an unfinished design), (4) build the model based

on the obtained data, analyze and verify it w.r.t. the conjectures, learn the consequences of the use of the developed model and make a new conjecture[5].

One of the distinguishing features of symbolic regression is the fact that model structure does not have to be imposed upfront, but is being discover through the evolutionary search. This implies that the use of symbolic regression in the iterative process of learning in a spirit of RSM adequately eliminates the conjecture phase of the conjecture-design-experiment-analysis loop. It is important to note that the goal of iterative learning with symbolic regression depicted in Figure 8.5 is somewhat broader than the goal of RSM, in the sense that RSM as depicted in Figure 8.4 (see (Box and Draper, 1986; Khuri and Cornell, 1987)) aims at gathering knowledge through confirming or denying conjectures, whereas symbolic regression aims at gathering knowledge in general. Specifically, the main goals of the framework of adaptive data collection with symbolic regression introduced in (Kotanchek et al., 2008) are:

- Modelling the response behavior,
- Identifying prediction uncertainty of data-derived models,
- Reducing uncertainty and improving accuracy via targeted data collection.

By adaptive data collection via trustable symbolic regression we mean sequential data collection and modeling on *nested* designs. The basic tenet of this approach is to use ensembles of diverse models to iteratively collect data points at the areas of maximal ensemble disagreement, i.e. in the areas of maximal prediction uncertainty; see also (Box and Draper, 1986).

Application of the generic idea of targeted data collection through ensemble disagreement can be traced back to early 90's, used for classification through ensembles of neural networks, and so-called query algorithms (Freund et al., 1993; Krogh and Vedelsby, 1995; Seung et al., 1992). Probably, the introduction of multi-objective fitness functions which reinforced diversity into symbolic regression models, enabled the application of ensemble-based adaptive data collection for response surface identification.

[5]Our main focus is on the regression part of RSM, where the the response surface is being constructed in the region of interest. We do not touch the optimization part of RSM, where steepest descent or adapted steepest descent algorithms are used for searching for optimum. We refer the interested reader to (Angün et al., 2002; Kleijnen, 2008b)

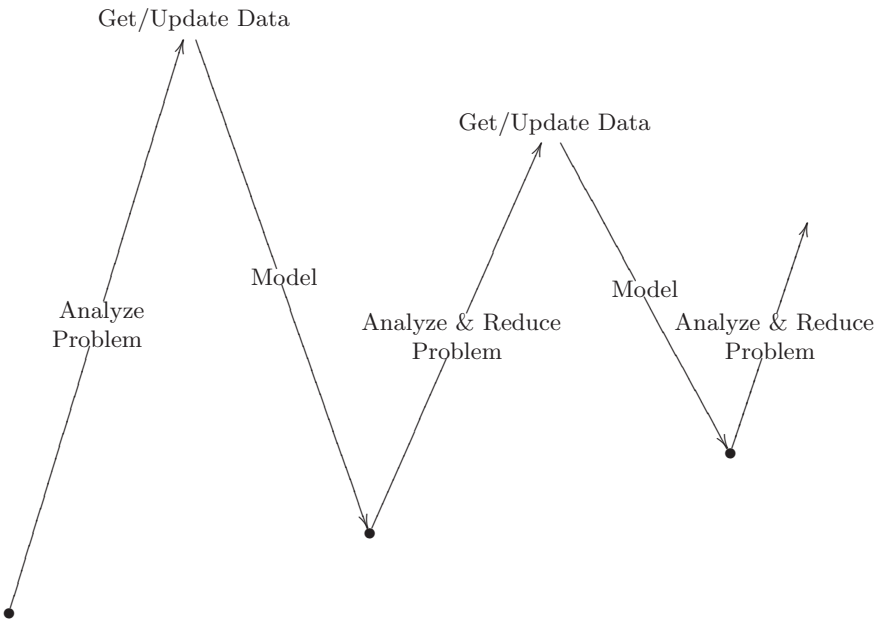


Figure 8.5: The benefit of ensemble-based adaptive experimentation via trustable symbolic regression is that it removes the Conjecture step from the learning process. Instead of having to blindly choose a potentially winning option from a multitude of options and make a conjecture about model structure and complexity before designing an experiment, the user is offered an opportunity to select from the set of winner options - Pareto GP solution ensembles. In addition, symbolic regression also provides the user with trustworthiness estimation of generated predictions.

We would like to emphasize that ensembles in this application of trustable symbolic regression are created for estimating uncertainty of prediction, and ensemble models are constrained to agree on all available data points and be diverse, in a sense that they disagree in predictions everywhere but in the data records. This approach is different from boosting, which combines models that are accurate on different subsets of data for improving the general prediction accuracy, and understands the model diversity in terms of diversity of the information captured by ensemble members.

The framework of targeted data collection presented in (Kotanchek et al., 2008) can be summarized in the following generic steps:

1. Build models on initial data collection and select an ensemble of diverse models and in parallel perform problem analysis and reduction,

2. Identify locations of maximal prediction uncertainty by maximizing the ensemble disconsensus function. (Extrema can be found using efficient numerical optimization algorithms for global or local optima, like NMinimize or FindMinimum functions of Mathematica.

3. Collect new data at appropriate locations (taking into account physical constraints for real measurements, or feasibility of simulated realizations for simulated data). (Kotanchek et al., 2008) demonstrates a hybrid strategy of collecting three new points per iteration of adaptive data collection - the choice is made to collect two new points, located at the maximum and minimum of ensemble prediction, and the third new point is located in the point of maximal ensemble disagreement, found at step (2). It might be beneficial to introduce some balancing strategies into the selection of new data records, such that in addition to exhibiting high ensemble disagreement they would also form space-filling sets when combined with the previously collected data.

4. Build new models (possibly performing variable selection) and repeat steps (1)-(3). (Since the new points contain a point of maximal ensemble disagreement, the old ensemble models are not likely to be valid or accurate on the new points, and therefore remodeling is required. Previous results can be seeded as the initial population into the new evolutionary run.)

This is a preliminary formulation of the proposed framework and further research is necessary to search for more appropriate selection schemes for model ensembles, and appropriate selection schemes for new locations of data records. However, the positive impact of the framework even in such a simplistic formulation, illustrated on several examples in (Kotanchek et al., 2008), is impressive, since it allows a *fully automatic* production of trustworthy global non-linear non-parametric regression models, coupled with a *fully automatic* data collection targeted at the areas of maximal uncertainty in the non-linear behaviour of the unknown system.

9 Conclusions

9.1 Summary of contributions

9.1.1 Contributions in view of research questions

This thesis has studied adaptive data-driven problem solving as an iterative series of the following steps: (1) acquisition, analysis and adaptation of data, (2) development of diverse collections of robust and interpretable models that explicitly describe the relationships of given data, and (3) analysis of the developed models for the purpose of obtaining insight into the data-generating system for its interpretation, analysis, and adaptation.

Having to work mostly with given multidimensional data, we studied ways to perform an assessment of data balancedness and overall quality in order to maximize the insight into the data *a priori* and independently from the modeling tool, and hence without modeling-bias. Chapter 2 attempts to resolve research question (6) of section 1.4.4 on the ways to balance given data for the purpose of improving the results of Pareto GP. It presents several intriguingly simple heuristics for weighting, ranking and compressing multi-dimensional input-output data and for inference of the data's cumulative information content.

For model development we use a particular method for creating symbolic regression models, called Pareto genetic programming. Pareto GP is an evolutionary strategy for searching for simple non-linear expressions of input-output relationships in the space of all symbolic expressions of variable length. In Pareto GP, selection of potentially suitable expressions is guided by more

than one criterion of fitness. Competing objectives like maximization of prediction accuracy and minimization of structural complexity of input-output expressions are optimized simultaneously in the course of a model development run. Pareto GP also has a memory in the form of an archive of 'best' expressions that is maintained and updated during the run. Pareto-aware archiving of successful intermediate solutions throughout the evolutionary run explicitly enforces structural and behavioral diversity of GP solutions and makes it possible to collect statistics on the presence of variables in good solutions and perform input sensitivity analysis during the run; see (Smits et al., 2005) and Chapter 8 of the thesis.

A Pareto GP system does not lack creativity (which is true for all evolutionary strategies), and even when it is constrained to produce accurate and concise expressions, it often generates compact models that agree with the data but show a wild behavior 'in-between' the data points. The risk of over-fitting, i.e. of unnecessary growth in the non-linearity without improvements in prediction accuracy, and over-specialization on training data at the expense of poor prediction on the test data, is the scourge of all (other than linear) empirical modeling methods and has to be avoided at any cost.

Research question (3) on avoidance of over-fitting in Pareto GP is tackled in Chapter 4. Experiments of that chapter bring us to the firm conclusion that explicit non-linearity control must be performed during the evolutionary run to avoid creation of senseless models. (Experiments with explicit control of expressional complexity of regression models clearly demonstrate that the risk of over-fitting is not eliminated, since response surfaces determined by compact expressions may still exhibit highly non-linear behavior.)

Non-linearity control can be done in many ways, such as interval arithmetics, numerical stability and range checking, catching pathologies on intermediate test sets, etc.; see, e.g.,(Keijzer, 2003; Kotanchek et al., 2007). The potential benefit of explicit control of the non-linearity complexity measure presented in Chapter 4 (performed independently from interval arithmetic calculations, and hence at additional costs), lies in the fact that it explicitly maintains the behavioral diversity of solutions w.r.t. non-linearity in addition to mitigating the risk of pathologies among best-of-the-run solutions.

Research questions (2) and (4) are addressed by introducing a heuristic for alternating multiple optimization objectives in a two-dimensional optimization

framework. Such alternating of objectives at each generation allows exploiting the effectiveness of two-dimensional optimization when more than two objectives are of interest (we considered prediction accuracy, expressional complexity, and the order of non-linearity). It therefore contributes to modified and seemingly improved navigation through the search space of potential solutions. When applied to alternation of expressional complexity and the order of non-linearity, it contributes to reduced over-fitting. Results of the experiments on all test problems of Chapter 4 show that alternating the order of non-linearity of GP individuals with their structural complexity in parallel with prediction accuracy maximization produces solutions that are both compact and have smoother response surfaces; hence, they contribute to better interpretability and easier deployment.

Research question (1) on the development of algorithmic improvements of the Pareto GP system, which would stimulate the creation of 'better' solutions faster, is one of the main goals of this thesis. It is addressed in Chapters 5 – 7. Questions (2), (3) and (5) are tackled in these Chapters as well.

It is important to emphasize that despite the fact that interesting results and significant performance improvements are obtained with respect to all research questions raised in this thesis, the questions cannot be discarded, and should be subjected to further research. The further research should not only aim at validating the proposed methods on a larger set of problems, or at improving them with respect to open questions mentioned in all Chapters of this thesis. The research should also aim at developing new and better methods to resolve the research questions of section 1.4.4 and to produce models satisfying the requirements of section 1.2.

9.1.2 Practical impact of this research

The growing demand for symbolic regression modeling within the R&D department of Dow Benelux B.V. and the Dow Chemical Company and the success stories which preceded and even accompanied this study, have been encouraging to further pursue the research.

This PhD project contributed to:

- The development of a prediction model for characterization of low-density polyethylenes, used for on-line and off-line process control, implemented on

three plants, running for already two years and recognized in 2007 as the most effective technology for characterizing materials of this group [1].

- The implementation and testing of a novel algorithm for creating input-output (structure-property) relationships, patented by the Dow Chemical Company.

- The minor and major enhancements of the GP toolbox for MatLab utilized by researchers of the Dow Benelux B.V. and the Dow Chemical Company in nearly 30 applications [2], and used for model building in 18 internal reports and several external publications, among which are (Castillo et al., 2006; Jordaan et al., 2004; Kordon et al., 2004; Kordon and Lue, 2004; Kordon et al., 2006) [3].

9.2 Application Domain of Symbolic Regression

Application of symbolic regression together with classical modeling techniques may have a synergetic effect in both industry and academia. The potential areas of co-application of symbolic regression with other data-driven modeling methods are mentioned in the introduction. We summarize the industrial applications, where symbolic regression via Pareto GP can have (or has had) a big impact. Some of these benefits are illustrated for application areas in manufacturing industries, but will also hold for other industries, e.g., finance, information technology, or services (see Chapter 1).

Integration of theoretical and empirical modeling

The ever-growing industrial competition demands shorter development cycles. When time is crucial, the empirical generation of fundamental relationships in process variables is certainly preferable over the discovery of first-principle models and developing a mathematical apparatus to efficiently operate with them. For this reason, this thesis studies the possibilities of efficient generation of diverse sets of plausible and transparent *data-driven* models in a framework of iterative

[1] The value and the information is kindly provided by Sjoerd de Vries, Dow Benelux B.V.
[2] The number of projects is kindly provided by Guido Smits, my co-supervisor.
[3] This list of papers and reports is kindly provided by Arthur Kordon, The Dow Chemical Company

knowledge extraction and problem solving. When such a technology is developed and proliferates in industry, then both high-throughput empirical modeling cycles and the low-throughput fundamental modeling insights will take place at the same time, provide feed-back for each other, and synergetically contribute to understanding the system in question.

Chapter 1 states that understanding of the fundamental laws of nature, and hence, the development of fundamental models that would help to control or predict the consequences, is one of the main goals of industrial research. The problem is that in some profit oriented companies, understanding the fundamental laws of nature is not the primary goal, but rather the secondary goal that would eventually lead to the primary goal of maximizing the profit. Especially nowadays, when a recession is being forecasted, when these companied take decisions on how to invest $\$M$ millions into k out of K ($K >> k$) R&D projects to maximize the profits in the coming months,the projects aimed at understanding fundamental models have little chance of being selected, since short-term results, and especially short-term profits are difficult to guarantee. I see and recommend empirical modelling through symbolic regression as a tool that would help 'fundamental modelers' in industry to cope with deadlines on deliveries of short-term results (which would be data-driven), while enhancing the potential and the delivery of long-term fundamental discoveries (which would be possible if the projects are funded and thinking time is avalable).

Research acceleration

Fast large-scale symbolic regression is a bridge between theoretical models and the available data. If sufficient speed can be achieved in discovering diverse insightful and applicable models describing complex multi-variate relationships on large-scale data, it will free more thinking time for researchers to develop fundamental models and perform intelligent experimentation. This will accelerate research, since the usual Data-Model-Insight cycle will be performed fast and will guide the development of fundamental models.

The unique capabilities of symbolic regression in automatic identification of driving variables and variable transformations allow an efficient reduction of the dimensionality and non-linearity of the problem. Since the problem space can be substantially reduced and 'linearized', new opportunities open well developed classical approximation techniques, since they can be applied to a much wider

range of problems. The novel approach to compressing the input-output data introduced in Chapter 2 allows to extract meaningful subsets of the data with the same or similar information content and can make the problem tractable for the modeling techniques that can produce accurate and robust models on small data sets (few thousands of records), but may fail in convergence on larger data sets.

The other applications of symbolic regression that industry can and has benefitted from are building inferential sensors, empirical emulators, modeling structure-properties relationships for new products and planning adaptive targeted experimentation.

Inferential sensors

Inferential sensors have been used in industry for quick predictions of difficult-to-measure process variables through combinations of available and easy-to-measure inputs. Inferential sensors generated with symbolic regression are widely used for controlling dynamic on-line processes; see, e.g., (Jordaan et al., 2004; Kordon et al., 2004; Kordon and Smits, 2001).

Emulators and Meta-Models

Empirical emulators are models mimicking the behavior of complex systems, either physical or already modelled, e.g. expensive black-box simulations. Emulators are often necessary for interpreting, adapting and optimizing the system in question. Transparency of models generated by symbolic regression makes them the first choice for building emulators of complex systems; see (Ashour et al., 2003; Kordon et al., 2003; Toropov and Alvarez, 1998) for examples.

Development of new products

Developing new products is an important application area for symbolic regression. Development of input-output models for discovery of structure-property or structure-activity relationships is a one of the major applications in the areas of drug discovery and polymer product development. Pharmaceutical companies are concerned with understanding complex multi-variate relationships between the molecular structures of the drug's compounds and the drug's activity. Polymer

research is looking for relationships between molecular structures of polymer mixtures and physical properties of the resulting polymers, like robustness, odour, and gloss. Both industries are now undergoing a shift towards high-throughput experimentation, where trustworthy symbolic regression is going to have a sweet spot (see the next section for elaboration on this statement).

9.3 Considerations on the future

The future of symbolic regression is undoubtedly bright. In addition to developing algorithmic improvements for symbolic regression via Pareto GP, this thesis makes a modest attempt to join the efforts of researchers practicing symbolic regression to popularize the technology as an eminent research acceleration and knowledge discovery tool.

It appears that despite all the benefits mentioned above and supported by industrial applications, the potential of symbolic regression via genetic programming is heavily underestimated by the modern research communities—at least in the areas of data mining and knowledge discovery within machine learning. The fraction of research papers aiming at scalable trustworthy symbolic regression is almost negligible in the total amount of publications. Supervised learning is mainly used for classification and only to a small extent for regression. Parametric approximations with neural networks still seem to be dominating regression-type applications. When I asked my colleagues why the efforts on classification and regression are so disproportional in publications and books on data mining, and why symbolic regression via genetic programming has not emerged as a leading application for regression in data mining, the answer was very interesting and quite prosaic. The colleagues hypothesized that it is mainly due to the fact that the technology is not implemented as a function in SAS and other popular software packages. I have little doubt that this "sin of omission" will be fixed in the nearest future, not only because symbolic regression automatically discovers trustable relationships on given data and automatically identifies significant variables and variable transformations, but also because in the last three years I have been observing the (exponentially) growing interest for this particular technology in industry (not only at Dow Chemical)[4].

[4]The scepsis of academia towards symbolic regression via GP also has seemed to be decaying if not exponentially, than still in a faster than linear fashion.

Of course, to make the future brighter, we will have to put considerable effort into making symbolic regression scalable and intuitively transparent for industrial applications. I am not stating that we have arrived at the point where symbolic regression is an industrial-scale technology. More efforts are needed to enable our tools to model data of hundreds of records in a minute, a million of records in an hour, or tens of thousands of variables in a day. I am stating that all necessary ingredients are there to make symbolic regression industrially scalable in the nearest future (maybe one-two years?). Recent advances in GPU computing, effective representations of individuals, model selection and niching strategies, co-operations, co-evolutions, multi-processor machines are all contributing conditions to pursue the research in trustworthy and scalable symbolic regression.

To support these statements, I would like to mention two reports that have been very encouraging, namely a "Roadmap for High Throughput Technologies" by InsightFaraday (2004) and the "Towards 2020 Science" report by Emmott et al. (2006).

According to (InsightFaraday, 2004), data handling and interpretation is one of the core capabilities required to underpin all high technology platforms, which can benefit from high-throughput experimentation. Identification of the key response parameters in data analysis and the ability to estimate them mathematically are stated as keys for developing robust process scale-up (InsightFaraday, 2004). In the mean time, these are the main distinctive capabilities of symbolic regression. In Table 9.1 we reproduce a fragment summarizing the required capabilities in data handling, interpretation, modeling and prediction. The features, which are required for high-tech platforms and by coincidence are currently present in symbolic regression via Pareto GP, are emphasized with a check mark. We recognize that the application of these capabilities to high-throughout technologies via symbolic regression is still a subject of further investigation. With an exclamation mark we denote the capabilities which would require minor to medium efforts to be implemented in symbolic regression, and therefore should be prioritized in future research.

In addition to data handling and interpretation, the other important point of application of symbolic regression to high-throughput research is in the automation and robotics engineering, in particular, real-time multi-parameter monitoring and full sensor feed-back for process control.

Table 9.1: Data Handling and Interpretation is one of the five cross-platform High Throughput capabilities Required to under-pin all technology platforms. The other four are Instrumentation, Automation and Robotics engineering, Integration, and Organizational Implementation (InsightFaraday, 2004) (extract of the table reproduced with permission of InsightFaraday Partnership). The required capabilities, which are currently present in symbolic regression via Pareto GP are emphasized with a check mark (note, that application of these capabilities to high-throughout technologies is still a subject to further investigation). With exclamation mark we emphasize the capabilities, which would require minor to medium efforts to implement in SR, and, therefore, they should be prioritized in the future research.

	Short term	*Medium term*	*Long Term*
Data Handling & Interpretation - Statistical Experimental design - Data capture and analysis - Data modeling and prediction - Data systems for operations - Process modelling and simulation	- Greater training in DoE: more access to statisticians ✓ Handling very large design spaces ✓ Multivariate modeling - Standards & Connectivity to allow cross-platform use	! User-friendly "intuitive" software ! "Live" data interpretation on large data sets ✓ Algorithms for inferential correlations ✓ Prediction for complex product performance ✓ Integrate scheduling into DoE and data interpretation feedback	✓ Cross-correlate numerous end-points effects ! Standards for data transfer: data streaming from remote networks ✓ Self-optimizing "whole-process" feedback algorithms

Another motivating challenge for application of symbolic regression lies in the phase of integration of high-throughput technology standards and operability, where the effective integration of design of experiments and data analysis is listed as the major issue. Experiments that can be performed in a high-throughput fashion, inevitably push the procedures on the design of these experiments and on modeling the obtained data towards automation. In particular, (InsightFaraday, 2004) states: *"...This is a particular issue in adoption of systems that allow for iterative feed-back - i.e. where integration of DoE and data analysis and data interpretation must be done "on the fly" - requiring methods for fast correlation in live data streams."* The framework for targeted data collection combined with the automatic trustworthy modeling seems to offer an intriguing possibility for high-throughput technologies in the integration of design of experiments, data analysis, and modeling in an automated fashion.

To conclude I would like to comment on the Draft Roadmap towards 2020 science, accompanying the "Towards 2020 Science" report edited by Emmott et al. (2006). This is an exciting thought-provoking document that represents selected directions of science between 2006 and 2020 with respect to five dimensions: Issues, Computational Platforms, Concepts/Tools from computer science, Scientific challenges, and Goals.

The Goals of this roadmap (like the system approach in biology, full model of a single cell, computational theory of synthetic biology) are more aligned with the goals of computational evolution suggested by Banzhaf et al. (2006). However, the 'conventional' artificial evolution, and particularly symbolic regression via GP, seem to be very relevant for the directions mentioned for scientific developments. Below we cite a subset of these developments, and speculate that the applicability of symbolic regression towards them has to be carefully and critically accessed, and, if positive, it should be used to actively guide future research.

Towards solution of Issues:
- Versioning of datasets, models, and algorithms used routinely and widespread in science.

Towards Computational Platforms:
- Move towards program execution for scientific applications hosted in the database (taking the application to the data, rather than the data to the application).
- Symbolic computation integrated into scientific databases and programming languages.

- Ability to rapidly build complex distributed systems from shared components that are trustworthy, predictable, reliable, scalable and extensible.
- Proof of concept for automated experimentation (2006).

Towards Concepts/Tools from Computer Science:

- Bayesian networks used to model effects of toxins in metabolic pathways *(E.V.:Symbolic regression may be an alternative*)
- Active learning techniques start to proliferate in science - towards autonomous experimentation [and understanding of the brain].
- Integration of sensors [environmental, physiological, chemical] and machine learning and data management - towards automated experimentation.
- Widespread *de novo* creation of models, theories and solutions (e.g. protein-drug candidates) from data using advanced machine learning.
- Autonomous experimentation and sophisticated sensor networks being used routinely to undertake otherwise impossible experiments.

Towards Scientific Challenges:

- 'Postdictive' modeling possible for a wide range of systems.

Towards Goals :

- Keystone-species identification.
- Reliable global warming, natural disaster and weather prediction models.
- Predictive models of effects of rainforest destruction, forest sustainability, effects on climate change on ecosystems, effects of climate change on foodwebs, restoration ecology planning, global health trends, sustainable agricultural solutions (2015).

I would like to repeat that these directions for scientific development seem to have a potential to benefit from the flexible principles of industrial scale symbolic regression, several of which are presented in this thesis. The only way to check whether trustworthy symbolic regression can contribute to advancing the science towards 2020, is to actually try to apply it to the corresponding challenges in an organized fashion.

Over the last three years I have learned one thing for sure: the advancements in the modern algorithms are guided by the challenges we are trying to meet. Predictive and computational powers of evolutionary modeling methods are literally co-evolving together with the complexity of the new modeling problems. This process does not show any signs of stagnation in the nearest future,

which means that the future of computational scientists is bright: there is plenty of work, with success almost guaranteed, irrespectively of economic downturns. So may the bond of an open mind and the adventurous inclination towards combining working principles of evolution into meta-principles, meta-strategies, and meta-heuristics, bring us the sweet taste of convergence to the accomplishment of our goals.

A

Chebyshev Polynomial Approximations

Approximating a function means finding for a given function f a function g from a certain class that is in a specific way close to f. There exist a great number of approximation problems depending on the class of functions where g is being sought from, the method by which g is being found, and the meaning of closeness between g to f.

Interpolation of a function is a partial case of the approximation problem, when values of g must coincide with the values of f in some points. These points are called knots of interpolation, or interpolation nodes.

To evaluate the closeness of the given function f and its approximation g the metrics of various functional spaces are used depending on a problem. Usually these are the metrics of the space of functions continuous on an interval (finite or infinite, also multidimensional), $C([a,b])$, and functions integrable up to an order p on an interval, $L_p([a,b])$. Metrics are the measures of distance between functions. For the above mentioned spaces they are determined by formulae:

$$\|f-g\|_{C([a,b])} = \max_{x\in[a,b]} |f(x)-g(x)|,$$

$$\|f-g\|_{L_p([a,b])} = (\int_{[a,b]} |f(x)-g(x)|^p)^{1/p}.$$

Usually one looks for approximations of f in a form of

$$Pf = \sum_{j=1}^{n} \alpha_j \; g_j, \tag{A.1}$$

where $g_1, g_2, \ldots, g_n$ are given functions from a certain class.

Example 1. *Approximation by algebraic polynomials in $C([a,b])$:* f is a real continuous function on $[a,b]$, functions $g_j(x)$ are monomials x^j, $j = 1, \ldots, n$. Approximation $P(x) = \sum_{j=1}^{n} \alpha_j \; x^j$.

Example 2. *Approximation by trigonometric polynomials in $C([a,b])$:* f is a real continuous function on $[a,b]$, functions $g_j(x)$ are $\beta_i \cos ix + \gamma_i \sin ix$.

Example 3. *Lagrange interpolation in $C([a,b])$:* Given $f \in C([a,b])$, and a set of points $a = x_1 < x_2 < \cdots < x_n = b$, let $f(x_1), \ldots, f(x_n)$ be known. If $g_i(x) \equiv l_i(x) = \prod_{j,j\neq i} \frac{x-x_i}{x_j-x_i}$, and $\lambda_i(f) = f(x_i), i = 1, \ldots, n$, then the problem of linear interpolation can be formulated as follows:

For function f given in points $\{x_i\}, i = 1, \ldots, n$ find an approximation $L(x) = (Pf)(x) = \sum_i \alpha_i l_i(x)$, such that it coincides with f in points $\{x_i\}$: $L(x_i) = f(x_i)$ for all $i = 1, \ldots, n$. Points $\{x_i\}$ are called knots of interpolation, functions $l_i(x)$ - Lagrange monomials.

When f is given in a tabulated form $(x_1, f(x_1), \ldots, x_N, f(x_N))$ direct interpolation through $f(x_1), \ldots, f(x_N)$ is the worst way to approximate f, since no values are given between the points, and the quality of approximation cannot be evaluated. According to the Weierstrass approximation theorem, any continuous function on [a,b] can be approximated with any precision by a polynomial of a sufficiently large degree. This means that if f contains errors, a perfect approximation of f can be found that will model both f and the noise extremely well. When this happens, f is said to be *over-fitted*. Over-fitting should be avoided as much as possible since the error in tabulated values can be drastically magnified by the interpolating polynomial between the interpolation nodes and the approximation will then become useless. Instead, a smoothing of data with a *best fit* polynomial of *minimal degree* is encouraged.

The concept of the best fit polynomial, or polynomial of the best approximation is accredited to the Russian mathematician Pafnuty Lvovich Chebyshev. He studied the best approximations of continuous functions, and got a whole series of results on best fit polynomials.

If $\mathcal{P}_n$ is a space of algebraic polynomials of degree n on interval $[a,b]$, a polynomial of the best approximation of f in P_n in metric C is a polynomial $Pf \in \mathcal{P}_n$ with the least deviation from f, i.e.

$$\|Pf - f\|_{C([a,b])} = \max_{Q\in\mathcal{P}_n} \|f - Q\|_{C([a,b])} \equiv E_n(f). \tag{A.2}$$

Chebyshev proved that for given degree n the best approximating polynomial Pf for function f in $C([a,b])$ is unique and is sufficiently characterized by the fact the the number of points on interval $[a,b]$ in which the difference $f(x) - Pf(x)$ takes the value $\max_{x\in[a,b]} |f(x) - Pf(x)|$ with alternating signs is at least $n+2$ (the celebrated Chebyshev alternation theorem (Rivlin, 1974)).

The main goal in approximating a continuous function by polynomials is a polynomial which among all polynomials of the same degree has a smallest deviation from the true function. It is known how to construct this polynomial for approximations in the space of square-integrable functions L_2. However, for other spaces the construction of the best fit polynomial is a difficult problem, with a solution known only for a few single cases. Therefore, several techniques have been developed to build 'nearly' best fit approximations.

For major classes of functions studied in calculus there exists such polynomial approximations for which approximation error decreases at the same order as the error of the best fit polynomial by $n \to \infty$. Examplea are an approximation of a periodic function by partial sums of its Fourier series, or approximation of a continuous function by trigonometric interpolation polynomials at equidistant knots, or algebraic interpolation polynomials with knots at zeros of Chebyshev polynomials.

Let us come back to the Example 3 and analyse the error of approximation of function f by the Lagrange interpolating polynomial L_n. For functions n times continuously differentiable on $[a,b]$, the following representation holds:

Theorem. Let $f \in C^{(n)}([a,b])$, and L_n be a Lagrange polynomial of degree $n-1$ constructed on points $a \leq x1, \ldots, x_n \leq b$. Then for any $x \in [a,b]$ there exist $\xi = \xi(x) \in [min(x_1, \ldots, x_n, x), max(x_1, \ldots, x_n)]$ such that

$$f(x) - L_n(x) = (x - x_1) \ldots (x - x_n) \frac{f^{(n)}}{n!}. \tag{A.3}$$

Thus, the difference of the function and its interpolating polynomial of degree $n-1$ can be represented as a polynomial of degree n. Let us denote it by $R_n(x)$. From (A.3) the remainder term can be estimated as:

$$\|R_n(x)\|_{C([a,b])} \leq \frac{\|f^{(n)}\|_{C([a,b])}}{n!} \|(x - x_1) \ldots (x - x_n)\|_{C([a,b])}, \tag{A.4}$$

where $\|g\|_{C([a,b])} = \max_{x\in[a,b]} |g(x)|$.

The bound in equation (A.4) for Lagrange interpolation is minimized if polynomial $P_n(x) = (x - x_1)\dots(x - x_n)$ has minimal deviation from zero, i.e. has minimal norm $\|P_n\|_{C([a,b])}$. Such a polynomial is unique for a fixed n. It carries the name Chebyshev; see (Rivlin, 1974; Tchebycheff, 1857).

Chebyshev polynomials denoted by $T_n(x)$, $n \geq 0$ are determined recursively:

$$T_0(x) = 1,\ T_1(x) = x, \tag{A.5}$$

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad n > 1. \tag{A.6}$$

The consequence of (A.6) is that the leading term of $T_n(x)$ is 2^{n-1}.

For $x \in [-1, 1]$ the following representation holds: $T_n(x) = \cos(n \arccos x), n \geq 0,\ x \in [-1, 1]$. Therefore,

1)$|T_n(x)| \leq 1, / > n \geq 0, > x \in [-1, 1]$,

2)T_n has $n + 1$ extrema on [-1,1]: $\hat{x}_m = \cos \frac{\pi m}{n}$, $m = 0, \dots, n$, and $T_n(\hat{x}_m) = (-1)^m$.

3)T_n has n distinct zeros: $x_m = \cos\left(\frac{\pi(2m-1)}{2n}\right) \in [-1, 1]$, $m = 1, \dots, n$.

Since n distinct zeros of a polynomial $T_n(x)$ of degree n are found, it can be represented as $T_n(x) = 2^{n-1}(x - x_1)\dots(x - x_n)$.

4) Polynomials $T_0, T_1, \dots, T_{n-1}$ are linearly independent. Moreover, in $L_2([a, b])$ they are orthogonal with a weighting coefficient $\frac{1}{\sqrt{1-x^2}}$:

$$\int_{-1}^{1} \frac{T_i(x)T_j(x)}{\sqrt{1 - x^2}} = \frac{\pi}{2}\delta_{ij}, \quad i, j = 1, \dots, n. \tag{A.7}$$

Besides, the system of Chebyshev polynomials satisfies a discrete orthogonality relation. If x_m, $m = 1, \dots, n$ are n zeros of polynomial$T_n(x)$, then for $0 < i, j < n$

$$\sum_{m=1}^{n} T_i(x_m)T_j(x_m) = \frac{m}{2}\delta_{ij}, \tag{A.8}$$

$$\sum_{m=1}^{n} T_0^2(x_m) = m. \tag{A.9}$$

The most important property of polynomial $T_n(x)/2^{n-1}$ is that it has minimal norm among all polynomials $P_n(x)$ with the leading coefficient equal to 1

(*miminax property*), i.e.

$$\max_{x\in[-1,1]} |P_n(x)| \geq \max_{x\in[-1,1]} |T_n(x)/2^{n-1}| = 1/2^{n-1}. \tag{A.10}$$

Polynomial $T_n(x)/2^{n-1}$ is thus called a polynomial with minimum deviation from zero on [-1,1].

Introducing a transform $z : [a, b] \to [-1, 1]$ such that $z(x) = (2x-(b+a))/(b-a)$, we can determine a Chebyshev polynomial on interval $[a, b]$:

$$T_n(x; a, b) = T_n(z(x)) = T_n((2x - (b + a))/(b - a)). \tag{A.11}$$

$\hat{T}_n(x)$ has n zeros on $[a, b]$, and leading coefficient $(2/(b-a))^n \cdot 2^{n-1} = 2^{2n-1}/(b-a)^n$.

The minimax property (A.10) implies that polynomial $\tilde{T}(x) = T_n(x; a, b)(b-a)^n/2^{2n-1}$ has a minimal norm among all polynomials P_n on $[a, b]$ with leading coefficient 1, i.e.

$$\max_{x\in[a,b]} |P_n(x)| \geq \max_{x\in[a,b]} |\tilde{T}_n(x)| = (b - a)^n/2^{2n-1}. \tag{A.12}$$

Now after a polynomial of degree n with minimum deviation from zero on interval $[a, b]$ is constructed, the error bound (A.4) can be minimized. Let select $P_n(x) = (x - x_1) \ldots (x - x_n) = \tilde{T}(x) = (b-a)^n/2^{2n-1} T_n((2x - (b+a))/(b-a))$. By definition of $P_n(x)$, the points $x_1, \ldots, x_n$ are zeros of polynomial $\tilde{T}_n(x)$ on interval [a,b], where

$$x_m = \frac{a+b}{2} + \frac{b-a}{2} \cos(\pi \frac{2m-1}{2n}), \quad m = 1, \ldots, n. \tag{A.13}$$

From (A.12) $\|P_n\|_C([a, b]) = (b - a)^n/2^{2n-1}$. Hence, if zeros (A.13) are selected as nodes for Lagrange interpolation of degree $n - 1$ for function $f \in C^{(n)}([a, b])$, the upper bound of the interpolation error is:

$$\|R_n(x)\|_{C([a,b])} \leq \frac{\|f^{(n)}\|_{C([a,b])}}{n!}(b - a)^n/2^{2n-1}. \tag{A.14}$$

The given bound for Lagrange interpolation over zeros of Chebyshev polynomial of degree n on [a,b] cannot be improved.

The exact characteristics of the behavior of the error of Lagrange interpolation

of degree $n-1$ over zeros (A.13)of Chebyshev polynomial $\tilde{T}_n$ is given by the following bound:

$$E_{n-1}(f) \leq \|f - L_n\|_{C([a,b])} \leq (2 + \frac{2}{3}\ln n)E_{n-1}(f), \tag{A.15}$$

where $E_{n-1}(f)$ is the error or the best-fit polynomial for f on $[a,b]$ defined in equation (A.2).

This bound shows that interpolation of order $n-1$ over zeros of Chebyshev polynomial $\tilde{T}_n$ gives an approximation that does not differ much from the best fit polynomial. This gives a so called rule of thumb: in polynomial approximation the 'best *a priori*' choice is the Lagrange interpolation at zeros of the Chebyshev polynomial.

Implementation details

The approximating polynomial can be found as an expansion over Chebyshev polynomials on $[a,b]$:

$$P(x) = \sum_{i=0}^{n-1} \alpha_i T_i(x;a,b), \tag{A.16}$$

$$T_i(x;a,b) = T_i(\frac{2x-b-1}{b-a}), \quad i = 0,1,\ldots,n-1. \tag{A.17}$$

It is required that $P(x)$ satisfies the following condition: for a given set of points $x_1,\ldots,x_n$

$$\sum_{k=1}^{n} T_i(x_k;a,b)f(x_k) = \sum_{k=1}^{n} T_i(x_k;a,b)P(x_k), \quad i = 0,\ldots,n-1. \tag{A.18}$$

By choosing points $x_1,\ldots,x_n$ to be zeros of Chebyshev polynomial $T_n(x;a,b)$ on $[a,b]$, we can find coefficients (α_i) of the expansion $P(x) = \sum_{i=0}^{n-1} \alpha_i T_i(x;a,b)$:

$$\alpha_i = \frac{\sum_{k=1}^{n} T_i(x_k;a,b)f(x_k)}{\sum_{k=1}^{n} T_i(x_k;a,b)T_i(x_k;a,b)}. \tag{A.19}$$

The approximation determined in this way can be constructed with low computational efforts. Condition (A.18) does not require the approximation $P(x)$ to coincide with the function $f(x)$ in any point on $[a,b]$. However,

wonderful properties of Chebyshev polynomials ensure that $P(x)$ will coincide with the Lagrange interpolating polynomial, built over zeros of n-th Chebyshev polynomial $T_n(x;a,b)$ on $[a,b]$. This means that $P(x)$ at first has equal values as $f(x)$ in n points defind in equation (A.13), and also minimizes the error of interpolation of f on all polynomials of degree $n-1$.

Coefficients $(\alpha)_i$ are computed from the following set of equations:

$$\begin{aligned}
\alpha_0 &= 1/n(T_0(x_1;a,b)f(x_1) &&+\cdots+ T_0(x_n;a,b)\ f(x_n))\\
\alpha_1 &= 2/n(T_1(x_1;a,b)f(x_1) &&+\cdots+ T_1(x_n;a,b)\ f(x_n))\\
&\vdots &&\vdots \qquad\qquad \vdots\\
\alpha_{n-1} &= 2/n(T_{n-1}(x_1;a,b)f(x_1)+\cdots+T_{n-1}(x_n;a,b)f(x_n))
\end{aligned} \tag{A.20}$$

The system of Chebyshev polynomials on thr interval $[a,b]$ satisfies a recurrent relation:

$$T_0(x;a,b)=1, T_1(x;a,b)=\frac{2x-(b+a)}{b-1},$$

$$T_{n+1}(x;a,b)=2\frac{2x-(b+a)}{b-1}T_n(x;a,b)-T_{n-1}(x;a,b). \tag{A.21}$$

Columns of table (A.20) for coefficients (α_i) should be computed from left to the right, in a top-down manner.

Let us denote $T_i(x_j;a,b)f(x_j)=g_{i,j}, z_j=2\cos\frac{\pi(2j-1)}{2n}$, $i=0,\ldots,n-1, j=1,\ldots,n$.

Then the $g_{i,j}$ are computed by recurrent formulae (A.21):

$$g_{0,j}=f(x_j),\quad g_{1,j}=1/2z_jf(x_j)=1/2z_jg_{0,j},$$

$$g_{i,j}=z_jg_{i-1,j}-g_{i-2,j},\quad i=2,\ldots,n-1. \tag{A.22}$$

As soon as a column of the table is computed, it is added to the sum of previous columns:

$$\alpha_i:=\alpha_i+g_{i,j},\qquad i=0,\ldots,n-1. \tag{A.23}$$

At the next step the coefficients α_i are computed as

$$\alpha_0 := 1/n\alpha_0, \quad \alpha_i := 2/n\alpha_i, \quad i = 1, \ldots, n-1. \tag{A.24}$$

Now, when coefficients of $P(x) = \sum_{i=0}^{n-1} \alpha_i T_i(x; a, b)$ are found, we can evaluate the quality of approximation. One way to do this would be to evaluate each $T_i(x; a, b)$ in a set of point where the error is sought for, while accumulating the sum $P(x)$. Instead, we are using the Clenshaw recurrence relation, which performs both processes simultaneously (Press et al., 1992). Applied to the Chebyshev recurrence in the form (A.21) and an approximation $P(x)$, the Clenshaw formula looks like:

$$\begin{aligned} y_{n+1} &\equiv y_n \equiv 0, \\ y_p = 2\frac{2x-(b+a)}{b-1} y_{p+1} - y_{p+2} &+ \alpha_p, \\ p &= n-1 \ldots, 0 \end{aligned} \tag{A.25}$$

By expressing (α_i) from (A.25) and substituting them in (A.16) one gets an efficient way to compute values of P by evaluating recurrence relation (A.25) without finding values of the Chebyshev system:

$$P(x) = \alpha_0 - y_2 + \frac{2x-(b+a)}{b-1} y_1. \tag{A.26}$$

Here x is a vector of points, which implies that (A.25) should only be computed once to find the values of $P(x)$ and, hence, the error of approximation in x.

Bibliography

Aggarwal, C. C., Hinneburg, A., and Keim, D. A. 2001. On the surprising behavior of distance metrics in high dimensional spaces. In *ICDT '01: Proceedings of the 8th International Conference on Database Theory*. Springer-Verlag, London, UK, 420–434.

Aggarwal, C. C. and Yu, P. S. 2001. Outlier detection for high dimensional data. In *SIGMOD '01: Proceedings of the 2001 ACM SIGMOD international conference on Management of data*. ACM, New York, NY, USA, 37–46.

Ahluwalia, M. and Bull, L. 2001. Coevolving functions in genetic programming. *Journal of Systems Architecture 47,* 7, 573–585.

Angün, E., Kleijnen, J. P. C., Hertog, D. D., and Gürkan, G. 2002. Recent advances in simulation optimization: response surface methodology revisited. In *WSC '02: Proceedings of the 34th conference on Winter simulation*. Winter Simulation Conference, 377–383.

Ashour, A. F., Alvarez, L. F., and Toropov, V. V. 2003. Empirical modelling of shear strength of RC deep beams by genetic programming. *Computers and Structures 81,* 5 (Mar.), 331–338.

Banzhaf, W., Beslon, G., Christensen, S., Foster, J. A., Képès, F., Lefort, V., Miller, J. F., Radman, M., and Ramsden, J. J. 2006. From artificial evolution to computational evolution: A research agenda. *Nature Reviews Genetics 7*, 729–735.

Banzhaf, W. and Langdon, W. B. 2002. Some considerations on the reason for bloat. *Genetic Programming and Evolvable Machines 3,* 1 (Mar.), 81–91.

Banzhaf, W., Nordin, P., Keller, R. E., and Francone, F. D. 1998. *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann, San Francisco, CA, USA.

Becker, Y., Fei, P., and Lester, A. M. 2006. Stock selection : An innovative application of genetic programming methodology. In *Genetic Programming Theory and Practice IV*, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic

and Evolutionary Computation, vol. 5. Springer, Ann Arbor, Chapter 12, 315–334.

Becker, Y. L., Fox, H., and Fei, P. 2007. An empirical study of multi-objective algorithms for stock ranking. In *Genetic Programming Theory and Practice V*, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation. Springer, Ann Arbor, Chapter 14, 241–262.

Bernard, H. and Gerlach, S. 1998. Does the term structure predict recessions? the international evidence. *International Journal of Finance & Economics 3,* 3 (July), 195–215.

Blickle, T. 1996. Evolving compact solutions in genetic programming: A case study. In *Parallel Problem Solving From Nature IV. Proceedings of the International Conference on Evolutionary Computation*, H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. LNCS, vol. 1141. Springer-Verlag, Berlin, Germany, 564–573.

Blickle, T. and Thiele, L. 1994. Genetic programming and redundancy. In *Genetic Algorithms within the Framework of Evolutionary Computation (Workshop at KI-94, Saarbrücken)*, J. Hopf, Ed. Max-Planck-Institut für Informatik (MPI-I-94-241), Im Stadtwald, Building 44, D-66123 Saarbrücken, Germany, 33–38.

Bongard, J. C. and Lipson, H. 2005. Nonlinear system identification using coevolution of models and tests. *IEEE Trans. Evolutionary Computation 9,* 4, 361–384.

Box, G. E. P. and Draper, N. R. 1986. *Empirical Model-building and Response Surfaces.* John Wiley & Sons, Inc., New York, NY, USA.

Brameier, M., Hoffmann, F., Nordin, P., Banzhaf, W., and Francone, F. 1999. Parallel machine code 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. Vol. 2. Morgan Kaufmann, Orlando, Florida, USA, 1228.

Bucci, A., Pollack, J. B., and de Jong, E. D. 2004. Automated extraction of problem structure. In *GECCO (1)*, K. Deb, R. Poli, W. Banzhaf, H.-G.

Beyer, E. K. Burke, P. J. Darwen, D. Dasgupta, D. Floreano, J. A. Foster, M. Harman, O. Holland, P. L. Lanzi, L. Spector, A. Tettamanzi, D. Thierens, and A. M. Tyrrell, Eds. Lecture Notes in Computer Science, vol. 3102. Springer, 501–512.

Caplan, M. and Becker, Y. 2004. Lessons learned using genetic programming in a stock picking context. In *Genetic Programming Theory and Practice II*, U.-M. O'Reilly, T. Yu, R. L. Riolo, and B. Worzel, Eds. Springer, Ann Arbor, Chapter 6, 87–102.

Castillo, F., Kordon, A., Smits, G., Christenson, B., and Dickerson, D. 2006. Pareto front genetic programming parameter selection based on design of experiments and industrial data. In *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, M. Keijzer, M. Cattolico, D. Arnold, V. Babovic, C. Blum, P. Bosman, M. V. Butz, C. Coello Coello, D. Dasgupta, S. G. Ficici, J. Foster, A. Hernandez-Aguirre, G. Hornby, H. Lipson, P. McMinn, J. Moore, G. Raidl, F. Rothlauf, C. Ryan, and D. Thierens, Eds. Vol. 2. ACM Press, Seattle, Washington, USA, 1613–1620.

Cavaretta, M. J. and Chellapilla, K. 1999. Data mining using genetic programming: The implications of parsimony on generalization error. In *Proceedings of the Congress on Evolutionary Computation*, P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, Eds. Vol. 2. IEEE Press, Mayflower Hotel, Washington D.C., USA, 1330–1337.

Cherkassky, V. 2002. Model complexity control and statistical learning theory. *Natural Computing: an International Journal 1,* 1, 109–133.

Cherkassky, V. and Mulier, F. 1998. *Learning from Data: Concepts, Theory, and Methods*. John Wiley & Sons, Inc., New York, NY, USA.

Choi, K., Youn, B., and Yang, R.-J. 2001. Moving least squares method for reliability-based design optimization. See Langdon et al. (2002).

Curry, R. and Heywood, M. I. 2004. Towards efficient training on large datasets for genetic programming. In *17th Conference of the Canadian Society for Computational Studies of Intelligence*, A. Y. Tawfik and S. D. Goodwin, Eds. LNAI, vol. 3060. Springer-Verlag, London, Ontario, Canada, 161–174.

Curry, R., Lichodzijewski, P., and Heywood, M. I. 2007. Scaling genetic programming to large datasets using hierarchical dynamic subset selection. *IEEE Transactions on Systems, Man, and Cybernetics: Part B - Cybernetics 37,* 4 (Aug.), 1065–1073.

Cuyt, A., Ed. 1993. *Rational Approximation theory: A state of the art.* Kluwer, Dordrecht.

Cuyt, A. 2003. Recent applications of rational approximation theory: a guided tour. In *Proc. Intern. Conf. NACOM*, G. Psihoyios, Ed. Wiley, Weinheim, 50–52.

Cuyt, A. and Verdonk, B. 1985. Multivariate rational interpolation. *Computing 34*, 41–61.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. 2002. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Trans. Evolutionary Computation 6,* 2, 182–197.

Doherty, K., Adams, R., and Davey, N. 2004. Non-euclidean norms and data normalisation. In *ESANN*. d-side publications, Bruges, Belgium, 181–186.

Domingos, P. 1999. The role of occam's razor in knowledge discovery. *Data Mining and Knowledge Discovery 3,* 4, 409–425.

Drucker, H. 1997. Improving regressors using boosting techniques. In *ICML*, D. H. Fisher, Ed. Morgan Kaufmann, Nashville, TN, USA, 107–115.

Dueker, M. 1997. Strengthening the case for the yield curve as a predictor of U.S. recessions. *Review of Federal Reserve Bank of St. Louis*, 41–51.

Eggermont, J. and van Hemert, J. I. 2000. Stepwise adaptation of weights for symbolic regression with genetic programming. In *Proceedings of the Twelveth Belgium/Netherlands Conference on Artificial Intelligence (BNAIC'00)*, A. van den Bosch and H. Weigand, Eds. KUB, De Efteling, Kaatsheuvel, Holland, 259–266.

Eggermont, J. and van Hemert, J. I. 2001. Adaptive genetic programming applied to new and existing simple regression problems. In *Genetic Programming, Proceedings of EuroGP'2001*, J. F. Miller, M. Tomassini, P. L.

Lanzi, C. Ryan, A. G. B. Tettamanzi, and W. B. Langdon, Eds. LNCS, vol. 2038. Springer-Verlag, Lake Como, Italy, 23–35.

Emmott, S., Shapiro, E., Rison, S., Phillips, A., and Herbert, A., Eds. 2006. *Towards 2020 Science*. Microsoft Research.

Estrella, A. and Mishkin, F. S. 1996. The yield curve as a predictor of U.S. recessions. *Current Issues in Economics and Finance* 6 (June).

Ferreira, C. 2001. Gene expression programming: A new adaptive algorithm for solving problems. *Complex Systems 13,* 2, 87–129.

Ficici, S. G. and Pollack, J. B. 2001. Pareto optimality in coevolutionary learning. In *ECAL '01: Proceedings of the 6th European Conference on Advances in Artificial Life*. Springer-Verlag, London, UK, 316–325.

Folino, G., Pizzuti, C., and Spezzano, G. 2006. GP ensembles for large-scale data classification. *IEEE Transactions on Evolutionary Computation 10,* 5, 604–616.

Fonseca, C. M. and Fleming, P. 1995. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation 3,* 1, 1–16.

François, D., Wertz, V., and Verleysen, M. 2007. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering 19,* 7, 873–886.

Francone, F. D., Deschaine, L. M., Battenhouse, T., and Warren, J. J. 2004. Discrimination of unexploded ordnance from clutter using linear genetic programming. In *Late Breaking Papers at the 2004 Genetic and Evolutionary Computation Conference*, M. Keijzer, Ed. Seattle, Washington, USA.

Freund, Y. 1995. Boosting a weak learning algorithm by majority. *Information and Computation 121,* 2, 256–285.

Freund, Y. and Schapire, R. E. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. In *EuroCOLT*, P. M. B. Vitányi, Ed. Lecture Notes in Computer Science, vol. 904. Springer, Barcelona, Spain, 23–37.

Freund, Y., Seung, H. S., Shamir, E., and Tishby, N. 1993. Information, prediction, and query by committee. In *Advances in Neural Information Processing Systems 5, [NIPS Conference]*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 483–490.

Friedman, J. H. 1991. Multivariate adaptive regression splines. *The Annals of Statistics 19,* 1, 1–67.

Gagné, C., Schoenauer, M., Parizeau, M., and Tomassini, M. 2006. Genetic programming, validation sets, and parsimony pressure. In *Proceedings of the 9th European Conference on Genetic Programming*, P. Collet, M. Tomassini, M. Ebner, S. Gustafson, and A. Ekárt, Eds. Lecture Notes in Computer Science, vol. 3905. Springer, Budapest, Hungary, 109–120.

Garshina, N. and Vladislavleva, C. 2004. On development of a complexity measure for symbolic regression via genetic programming. In *Modeling report for Dow Benelux B.V.* Technische Universiteit Eindhoven, Eindhoven, the Netherlands.

Gathercole, C. and Ross, P. 1994. Dynamic training subset selection for supervised learning in genetic programming. In *Parallel Problem Solving from Nature III*, Y. Davidor, H.-P. Schwefel, and R. Männer, Eds. LNCS, vol. 866. Springer-Verlag, Jerusalem, 312–321.

Gilbert, R. J., Goodacre, R., Shann, B., Kell, D. B., Taylor, J., and Rowland, J. J. 1998. Genetic programming-based variable selection for high-dimensional data. In *Genetic Programming 1998: Proceedings of the Third Annual Conference*, J. R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. Riolo, Eds. Morgan Kaufmann, University of Wisconsin, Madison, Wisconsin, USA, 109–115.

Graves-Morris, P. 1981. Efficient reliable rational interpolation. *Lecture Notes in Mathematics 888*, 28–63.

Gustafson, S., Ekart, A., Burke, E., and Kendall, G. 2004. Problem difficulty and code growth in genetic programming. *Genetic Programming and Evolvable Machines 5,* 3 (Sept.), 271–290.

Hansen, L. K. and Salamon, P. 1990. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence 12,* 10, 993–1001.

Harding, S. and Banzhaf, W. 2007. Fast genetic programming on gpus. In *EuroGP*, M. Ebner, M. O'Neill, A. Ekárt, L. Vanneschi, and A. Esparcia-Alcázar, Eds. Lecture Notes in Computer Science, vol. 4445. Springer, Valencia, Spain, 90–101.

Harmeling, S., Dornhege, G., Tax, D., Meinecke, F., and Muller, K.-R. 2006. From outliers to prototypes: Ordering data. *Neurocomputing 69,* 13-15 (August), 1608–1618.

Haykin, S. 1994. *Neural Networks: A Comprehensive Foundation.* Macmillan College Publishing Company, Inc., New York, NY, USA.

Ho, Y.-C. 2000. Soft optimization for hard problems, computerized lecture via private communication/distribution.

Ho, Y.-C., Cassandras, C., Chen, C.-H., and Dai, L. 2000. Ordinal optimization and simulation. *Journal of the Operational Research Society 51*, 490–500.

Holland, J. H. 1975. *Adaptation in Natural and Artificial Systems.* MIT Press, Cambridge, MA, USA.

Horn, J. and Nafpliotis, N. 1993. Multiobjective Optimization using the Niched Pareto Genetic Algorithm. Tech. Rep. IlliGAl Report 93005, Urbana, Illinois, USA.

Hornby, G. S. 2006. ALPS: the age-layered population structure for reducing the problem of premature convergence. In *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, M. Keijzer, M. Cattolico, D. Arnold, V. Babovic, C. Blum, P. Bosman, M. V. Butz, C. Coello Coello, D. Dasgupta, S. G. Ficici, J. Foster, A. Hernandez-Aguirre, G. Hornby, H. Lipson, P. McMinn, J. Moore, G. Raidl, F. Rothlauf, C. Ryan, and D. Thierens, Eds. Vol. 1. ACM Press, Seattle, Washington, USA, 815–822.

Hu, J., Goodman, E., Seo, K., Fan, Z., and Rosenberg, R. 2005. The hierarchical fair competition (hfc) framework for sustainable evolutionary algorithms. *Evol. Comput. 13,* 2, 241–277.

Iba, H. 1999. Bagging, boosting, and bloating in 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. Vol. 2. Morgan Kaufmann, Orlando, Florida, USA, 1053–1060.

Iba, H., de Garis, H., and Sato, T. 1994. Genetic programming using a minimum description length principle. In *Advances in Genetic Programming*, K. E. Kinnear, Jr., Ed. MIT Press, Cambridge, MA, USA, Chapter 12, 265–284.

InsightFaraday, P., Ed. 2004. *A Roadmap for High Throughput Technologies*. InsightFaraday Partnership, Runcorn, Cheshire, UK.

Jin, H., Ooi, B. C., Shen, H. T., Yu, C., and Zhou, A. 2003. An adaptive and efficient dimensionality reduction algorithm for high-dimensional indexing. In *ICDE*, U. Dayal, K. Ramamritham, and T. M. Vijayaraman, Eds. IEEE Computer Society, Bangalore, India, 87–95.

Jin, R., Chen, W., and Simpson, T. 2000. Comparative studies of metamodeling techniques under multiple modeling criteria. Tech. Rep. 2000-4801, AIAA.

Johnson, R. A. and Wichern, D. W., Eds. 1988. *Applied multivariate statistical analysis*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Jordaan, E., Kordon, A., Chiang, L., and Smits, G. 2004. Robust inferential sensors based on ensemble of predictors generated by genetic programming. In *Parallel Problem Solving from Nature - PPSN VIII*, X. Yao, E. Burke, J. A. Lozano, J. Smith, J. J. Merelo-Guervós, J. A. Bullinaria, J. Rowe, P. T. A. Kabán, and H.-P. Schwefel, Eds. LNCS, vol. 3242. Springer-Verlag, Birmingham, UK, 522–531.

Jordaan, E. M. 2002. Development of Robust Inferential Sensors: Industrial application of support vector machines for regression. Ph.D. thesis, Eindhoven University of Technology, Eindhoven, the Netherlands.

Keijzer, M. 2002. Scientific Discovery using Genetic Programming. Ph.D. thesis, Danish Technical University, Lyngby, Denmark.

Keijzer, M. 2003. 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. LNCS, vol. 2610. Springer-Verlag, Essex, UK, 70–82.

Keijzer, M. and Foster, J. 2007. Crossover bias in genetic programming. In *Proceedings of the 10th European Conference on Genetic Programming*, M. Ebner, M. O'Neill, A. Ekárt, L. Vanneschi, and A. I. Esparcia-Alcázar, Eds. Lecture Notes in Computer Science, vol. 4445. Springer, Valencia, Spain.

Khuri, A. I. and Cornell, J. A. 1987. *Response Surfaces: Designs and Analyses*. Marcel Dekker, Inc., New York, NY, USA.

Kleijnen, J. and Sargent, R. 2000. A methodology for fitting and validating metamodels in simulation. *European Journal of Operational Research 120,* 1, 14–29.

Kleijnen, J. P. 2008a. Kriging metamodeling in simulation: A review. *European Journal of Operational Research*.

Kleijnen, J. P. C. 2005. Forty years of statistical design and analysis of simulation experiments (dase). In *WSC '05: Proceedings of the 37th conference on Winter simulation*. Winter Simulation Conference, 1–17.

Kleijnen, J. P. C. 2008b. *DASE: Design and Analysis of Simulation Experiments*. Springer, New York, NY, USA.

Kleijnen, J. P. C., Sanchez, S. M., Lucas, T. W., and Cioppa, T. M. 2005. State-of-the-art review: A user's guide to the brave new world of designing simulation experiments. *INFORMS Journal on Computing 17,* 3, 263–289.

Kohavi, R. and John, G. H. 1997. Wrappers for feature subset selection. *Artificial Intelligence: Special Issue on Relevance 97,* 1-2, 273–324.

Kordon, A., Jordaan, E., Chew, L., Smits, G., Bruck, T., Haney, K., and Jenings, A. 2004. Biomass inferential sensor based on ensemble of models generated by genetic programming. In *Genetic and Evolutionary Computation – GECCO-2004, Part II*, K. Deb, R. Poli, W. Banzhaf, H.-G. Beyer, E. Burke, P. Darwen, D. Dasgupta, D. Floreano, J. Foster, M. Harman, O. Holland, P. L.

Lanzi, L. Spector, A. Tettamanzi, D. Thierens, and A. Tyrrell, Eds. Lecture Notes in Computer Science, vol. 3103. Springer-Verlag, Seattle, WA, USA, 1078–1089.

Kordon, A., Kalos, A., and Adams, B. 2003. Empirical emulators for process monitoring and optimization. In *MED2003: Proceedings of the IEEE 11th Conference on Control and Automation*. IEEE, Rhodes, Greece, 111.

Kordon, A. and Lue, C.-T. 2004. Symbolic regression modeling of blown film process effects. In *Proceedings of the 2004 IEEE Congress on Evolutionary Computation*. IEEE Press, Portland, Oregon, 561–568.

Kordon, A. K., Smits, G., Jordaan, E., Kalos, A., and Chiang, L. 2006. Empirical models with self-assessment capabilities for on-line industrial applications. In *Proceedings of the 2006 IEEE Congress on Evolutionary Computation*, G. G. Yen, L. Wang, P. Bonissone, and S. M. Lucas, Eds. IEEE Press, Vancouver, 10463–10470.

Kordon, A. K. and Smits, G. F. 2001. Soft sensor development using genetic programming. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, Eds. Morgan Kaufmann, San Francisco, California, USA, 1346–1351.

Korns, M. F. 2006. Large-scale, time-constrained symbolic regression. In *Genetic Programming Theory and Practice IV*, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation, vol. 5. Springer, Ann Arbor, Chapter 16, 299–314.

Kotanchek, M., Smits, G., and Kordon, A. 2003. Industrial strength genetic programming. In *Genetic Programming Theory and Practice*, R. L. Riolo and B. Worzel, Eds. Kluwer, Chapter 15, 239–256.

Kotanchek, M., Smits, G., and Vladislavleva, E. 2006. Pursuing the Pareto paradigm tournaments, algorithm variations & ordinal optimization. In *Genetic Programming Theory and Practice IV*, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation, vol. 5. Springer, Ann Arbor, Chapter 12, 167–186.

Kotanchek, M., Smits, G., and Vladislavleva, E. 2007. Trustable symbolic regression models: Using ensembles, interval arithmetic and pareto fronts to develop robust and trust-aware models. In *Genetic Programming Theory and Practice V*, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation, vol. 6. Springer, Ann Arbor, MI, USA, Chapter 12, 203–222.

Kotanchek, M., Smits, G., and Vladislavleva, E. 2008. Adaptive design of experiments: Using symbolic regression via genetic programming to guide data collection and model development. In *Genetic Programming Theory and Practice VI*, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation. Springer, Ann Arbor.

Koza, J. R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, Cambridge, MA, USA.

Koza, J. R. 1994. *Genetic Programming II: Automatic Discovery of Reusable Programs.* MIT Press, Cambridge Massachusetts.

Krogh, A. and Vedelsby, J. 1995. Neural network ensembles, cross validation, and active learning. In *Advances in Neural Information Processing Systems*, G. Tesauro, D. Touretzky, and T. Leen, Eds. Vol. 7. The MIT Press, Cambridge, MA, USA, 231–238.

Kudryavtsev, V. 1995. On the automata funcional systems. *Discrete Mathematics and Applications 5,* 5 (August), 397–424.

Landry, J., Kosta, L. D., and Bernier, T. 2006. Discriminant feature selection by genetic programming: Towards a domain independent multi-class object detection system. *Journal of Systemics, Cybernetics and Informatics. 3,* 1.

Langdon, W. B., Cantú-Paz, E., Mathias, K. E., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M. A., Schultz, A. C., Miller, J. F., Burke, E. K., and Jonoska, N., Eds. 2002. *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, New York, USA, 9-13 July 2002.* Morgan Kaufmann.

Langdon, W. B. and Poli, R. 1998. Fitness causes bloat: Mutation. In *Proceedings of the First European Workshop on Genetic Programming*, W. Banzhaf, R. Poli, M. Schoenauer, and T. C. Fogarty, Eds. LNCS, vol. 1391. Springer-Verlag, Paris, France, 37–48.

Langdon, W. B. and Poli, R. 2002. *Foundations of Genetic Programming*. Springer-Verlag, Heidelberg, Germany.

Langdon, W. B., Soule, T., Poli, R., and Foster, J. A. 1999. The evolution of size and shape. In *Advances in Genetic Programming 3*, L. Spector, W. B. Langdon, U.-M. O'Reilly, and P. J. Angeline, Eds. MIT Press, Cambridge, MA, USA, Chapter 8, 163–190.

Lasarczyk, C., Dittrich, P., and Banzhaf, W. 2004. Dynamic subset selection based on a fitness case topology. *Evolutionary Computation 12,* 2, 223–242.

Lau, T. E. and Ho, Y.-C. 1997. Universal alignment probabilities and subset selection for ordinal optimization. *J. Optim. Theory Appl. 93,* 3, 455–489.

Laumanns, M., Thiele, L., Zitzler, E., and Deb, K. 2002. Archiving with guaranteed convergence and diversity in multi-objective optimization. See Langdon et al. (2002), 439–447.

Lemczyk, M. and Heywood, M. I. 2007. Training binary GP classifiers efficiently: A Pareto-coevolutionary approach. In *Proceedings of the 10th European Conference on Genetic Programming*, M. Ebner, M. O'Neill, A. Ekárt, L. Vanneschi, and A. I. Esparcia-Alcázar, Eds. Lecture Notes in Computer Science, vol. 4445. Springer, Valencia, Spain, 229–240.

Levin, D. 1998. The approximation power of moving least-squares. *Mathematical Computation 67,* 224, 1517–1531.

Liu, Y. and Yao, X. 2002. Learning and evolution by minimization of mutual information. In *PPSN VII: Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*. Springer-Verlag, London, UK, 495–504.

Liu, Y., Yao, X., and Higuchi, T. 2000. Evolutionary ensembles with negative correlation learning. *IEEE Transactions on Evolutionary Computation 4,* 4 (November), 380.

McPhee, N. F. and Miller, J. D. 1995. Accurate replication in genetic programming. In *Genetic Algorithms: Proceedings of the Sixth International Conference (ICGA95)*, L. Eshelman, Ed. Morgan Kaufmann, Pittsburgh, PA, USA, 303–309.

Neely, C. J. and Weller, P. A. 2003. Intraday technical trading in the foreign exchange market. *Journal of International Money and Finance 22,* 2 (Apr.), 223–237.

Neely, C. J., Weller, P. A., and Dittmar, R. 1997. Is technical analysis in the foreign exchange market profitable? A genetic programming approach. *The Journal of Financial and Quantitative Analysis 32,* 4 (Dec.), 405–426.

Neshatian, K., Zhang, M., and Johnston, M. 2007. Feature construction and dimension reduction using genetic programming. In *Australian Conference on Artificial Intelligence*, M. A. Orgun and J. Thornton, Eds. Lecture Notes in Computer Science, vol. 4830. Springer, 160–170.

Nordin, P. 1994. A compiling genetic programming system that directly manipulates the machine code. In *Advances in Genetic Programming*, K. E. Kinnear, Jr., Ed. MIT Press, Boston, USA, Chapter 14, 311–331.

Nordin, P. and Banzhaf, W. 1995. Complexity compression and evolution. In *Genetic Algorithms: Proceedings of the Sixth International Conference (ICGA95)*, L. Eshelman, Ed. Morgan Kaufmann, Pittsburgh, PA, USA, 310–317.

Okabe, A., Boots, B., Sugihara, K., and Chiu, S. N. 2000. *Spatial Tessellations: Concepts and applications of Voronoi diagrams*, 2nd ed. Probability and Statistics. Wiley, New York, NY, USA. 671 pages.

O'Neill, M. and Ryan, C. 2003. *Grammatical Evolution: Evolutionary Automatic Programming in a Arbitrary Language*. Genetic programming, vol. 4. Kluwer Academic Publishers.

Paris, G., Robilliard, D., and Fonlupt, C. 2001. Applying boosting techniques to genetic programming. In *Artificial Evolution 5th International Conference, Evolution Artificielle, EA 2001*, P. Collet, C. Fonlupt, J.-K. Hao,

E. Lutton, and M. Schoenauer, Eds. LNCS, vol. 2310. Springer Verlag, Creusot, France, 267–278.

Poli, R. 1996. Genetic programming for feature detection and image segmentation. In *Evolutionary Computing*, T. C. Fogarty, Ed. Number 1143. Springer-Verlag, University of Sussex, UK, 110–125.

Poli, R. 2005. Tournament selection, iterated coupon-collection problem, and backward-chaining evolutionary algorithms. In *Foundations of Genetic Algorithms 8*, A. H. Wright, M. D. Vose, K. A. De Jong, and L. M. Schmitt, Eds. Lecture Notes in Computer Science, vol. 3469. Springer-Verlag, Aizu-Wakamatsu City, Japan, 132–155.

Poli, R. and Langdon, W. B. 2005. Backward-chaining genetic programming. In *GECCO 2005: Proceedings of the 2005 conference on Genetic and evolutionary computation*, H.-G. Beyer, U.-M. O'Reilly, D. V. Arnold, W. Banzhaf, C. Blum, E. W. Bonabeau, E. Cantu-Paz, D. Dasgupta, K. Deb, J. A. Foster, E. D. de Jong, H. Lipson, X. Llora, S. Mancoridis, M. Pelikan, G. R. Raidl, T. Soule, A. M. Tyrrell, J.-P. Watson, and E. Zitzler, Eds. Vol. 2. ACM Press, Washington DC, USA, 1777–1778.

Poli, R., Langdon, W. B., and Dignum, S. 2007. On the limiting distribution of program sizes in tree-based genetic programming. In *Proceedings of the 10th European Conference on Genetic Programming*, M. Ebner, M. O'Neill, A. Ekárt, L. Vanneschi, and A. I. Esparcia-Alcázar, Eds. Lecture Notes in Computer Science, vol. 4445. Springer, Valencia, Spain.

Poli, R., Langdon, W. B., and McPhee, N. F. 2008. *A field guide to genetic programming*. Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`. (With contributions by J. R. Koza).

Powell, M. D. J. 1987. Radial basis functions for multivariable interpolation: a review. 143–167.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. 1992. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA.

Provost, F. 2000. Machine learning from imbalanced data sets 101 (extended abstract). In *AAAI2000 Workshop on Imbalanced Data Sets*, N. Japkowicz and Cochairs, Eds. AAAI Press, Austin, TX,USA.

Rennen, G. 2008. Subset selection from large datasets for kriging modeling. Tech. rep.

Rivlin, T. J. 1974. *The Chebyshev Polynomials.* Wiley, New York, NY, USA.

Rosca, J. 1996. Generality versus size in genetic programming. In *Genetic Programming 1996: Proceedings of the First Annual Conference*, J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, Eds. MIT Press, Stanford University, CA, USA, 381–387.

Rosin, C. D. and Belew, R. K. 1997. New methods for competitive coevolution. *Evol. Comput. 5,* 1, 1–29.

Ross, S. 1976. Arbitrage theory of capital asset pricing. *Journal of Economic Theory 13*, 341–360.

Sacks, J., Welch, W., Mitchell, T. J., and Wynn, H. 1989. Design and analysis of computer experiments. *Statistical Science 4*, 409–435.

Salazar Celis, O., Cuyt, A., and Verdonk, B. 2007. Rational approximation of vertical segments. *Numerical Algorithms 45*, 375–388.

Salustowicz, R. and Schmidhuber, J. 1997. Probabilistic incremental program evolution. *Evolutionary Computation 5,* 2, 123–141.

Schapire, R. E. 1990. The strength of weak learnability. *Machine Learning 5,* 2, 197–227.

Schmidt, M. D. and Lipson, H. 2006. Co-evolving fitness predictors for accelerating and reducing evaluations. In *Genetic Programming Theory and Practice IV*, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation, vol. 5. Springer, Ann Arbor, MI, USA, Chapter 17, 113–130.

Sephton, P. 2001. Forecasting recessions: can we do better on mars? *Federal Reserve Bank of St. Louis Review in Business* Mar, 39–49.

Seung, H. S., Opper, M., and Sompolinsky, H. 1992. Query by committee. In *Computational Learning Theory*. 287–294.

Sharpe, W. 1964. Capital asset prices: A theory of market equilibrium under conditions of risk. *Journal of Finance 19(3)*, 425–442.

Sherrah, J. R., Bogner, R. E., and Bouzerdoum, A. 1997. The evolutionary pre-processor: Automatic feature extraction for supervised classification using genetic programming. In *Genetic Programming 1997: Proceedings of the Second Annual Conference*, J. R. Koza, K. Deb, M. Dorigo, D. B. Fogel, M. Garzon, H. Iba, and R. L. Riolo, Eds. Morgan Kaufmann, Stanford University, CA, USA, 304–312.

Shyy, W., Tucker, P. K., and Vaidyanathan, R. 1999. Response surface and neural network techniques for rocket engine injector optimization. Tech. Rep. 99-2455, AIAA.

Simpson, T. W. 1998. Comparison of response surface and kriging models in the multidisciplinary design of an aerospike nozzle. Tech. Rep. TR-98-16.

Smits, G., Kordon, A., Vladislavleva, K., Jordaan, E., and Kotanchek, M. 2005. Variable selection in industrial datasets using pareto genetic programming. In *Genetic Programming Theory and Practice III*, T. Yu, R. L. Riolo, and B. Worzel, Eds. Kluwer, Ann Arbor, MI, USA.

Smits, G. and Kotanchek, M. 2004. Pareto-front exploitation in symbolic regression. In *Genetic Programming Theory and Practice II*, U.-M. O'Reilly, T. Yu, R. L. Riolo, and B. Worzel, Eds. Springer, Ann Arbor, MI, USA, Chapter 17, 283–299.

Smits, G. and Vladislavleva, E. 2006. Ordinal Pareto genetic programming. In *Proceedings of the 2006 IEEE Congress on Evolutionary Computation*, G. G. Yen, L. Wang, P. Bonissone, and S. M. Lucas, Eds. IEEE Press, Vancouver, Canada, 3114 – 3120.

Smits, G. F. 2001. *Genetic Programming Toolbox for the Dow Chemical Company. Internal Report.* The Dow Chemical Company, Terneuzen, the Netherlands.

Song, D., Heywood, M. I., and Zincir-Heywood, A. N. 2005. Training genetic programming on half a million patterns: an example from anomaly detection. *IEEE Transactions on Evolutionary Computation 9,* 3 (June), 225–239.

Soule, T. and Foster, J. A. 1998. Effects of code growth and parsimony pressure on populations in genetic programming. *Evolutionary Computation 6,* 4 (Winter), 293–309.

Soule, T., Foster, J. A., and Dickinson, J. 1996. Code growth in genetic programming. In *Genetic Programming 1996: Proceedings of the First Annual Conference*, J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, Eds. MIT Press, Stanford University, CA, USA, 215–223.

Soule, T. and Heckendorn, R. B. 2002. An analysis of the causes of code growth in genetic programming. *Genetic Programming and Evolvable Machines 3,* 3 (Sept.), 283–309.

Streeter, M. J. 2003. The root causes of code growth in genetic programming. In *Genetic Programming, Proceedings of EuroGP'2003*, C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, and E. Costa, Eds. LNCS, vol. 2610. Springer-Verlag, Essex, UK, 443–454.

Sun, P. and Yao, X. 2006. Boosting kernel models for regression. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*. IEEE Computer Society, Washington, DC, USA, 583–591.

Tchebycheff, P. 1857. Sur les questions de minima qui se rattachent à la représentation approximative des fonctions. *Mémoires Acad. Sci. St. Pétersbourg par divers savants 7*, 191.

Teller, A. and Andre, D. 1997. Automatically choosing the number of fitness cases: The rational allocation of trials. In *Genetic Programming 1997: Proceedings of the Second Annual Conference*, J. R. Koza, K. Deb, M. Dorigo, D. B. Fogel, M. Garzon, H. Iba, and R. L. Riolo, Eds. Morgan Kaufmann, Stanford University, CA, USA, 321–328.

Topchy, A. and Punch, W. F. 2001. Faster genetic programming based on local gradient search of numeric leaf values. In *Proceedings of the Genetic*

and Evolutionary Computation Conference (GECCO-2001), L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, Eds. 155–162.

Toropov, V. 1989. Simulation approach to structural optimization. *Structural Optimization 1,* 1, 37–46.

Toropov, V. V. and Alvarez, L. F. 1998. Application of genetic programming to the choice of a structure of multipoint approximations. In *1st ISSMO/NASA Internet Conf. on Approximations and Fast Reanalysis in Engineering Optimization*. Published on a CD ROM.

Van Barel, M. and Bultheel, A. 1990. A new approach to the rational interpolation problem. *J. Comput. Appl. Math. 32*, 281–289.

Vapnik, V. 1982. *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Vapnik, V. 1997. The support vector method. In *ICANN '97: Proceedings of the 7th International Conference on Artificial Neural Networks*. Springer-Verlag, London, UK, 263–271.

Vladislavleva, E., Smits, G., and den Hertog, D. 2008. Order of nonlinearity as a complexity measure for models generated by symbolic regression via Pareto genetic programming. *IEEE Transactions on Evolutionary Computation. In press.*.

Vladislavleva, E., Smits, G., and Kotanchek, M. 2007. Better solutions faster : Soft evolution of robust regression models in Pareto genetic programming. In *Genetic Programming Theory and Practice V*, R. L. Riolo, T. Soule, and B. Worzel, Eds. Genetic and Evolutionary Computation. Springer, Ann Arbor, Chapter 2, 13–32.

Vladislavleva, E. J. 2005. *Symbolic Regression via Genetic Programming, P.D.Eng. Thesis for Dow Benelux B.V.* Technische Universiteit Eindhoven, Eindhoven, the Netherlands.

Wolpert, D. H. 1992. Stacked generalization. *Neural Networks 5,* 2, 241–259.

Xie, H., Zhang, M., and Andreae, P. 2007. An analysis of depth of crossover points in tree-based genetic programming. In *2007 IEEE Congress on Evolutionary Computation*, D. Srinivasan and L. Wang, Eds. IEEE Computational Intelligence Society, IEEE Press, Singapore, 4561–4568.

Yu, J., Yu, J., Almal, A. A., Dhanasekaran, S. M., Ghosh, D., Worzel, W. P., and Chinnaiyan, A. M. 2007. Feature selection and molecular classification of cancer using genetic programming. *Neoplasia 9,* 4 (Apr.), 292–303.

Zhang, B.-T. 1999. Bayesian genetic programming. In *Foundations of Genetic Programming*, T. Haynes, W. B. Langdon, U.-M. O'Reilly, R. Poli, and J. Rosca, Eds. Orlando, Florida, USA, 68–70.

Zhang, B.-T. and Cho, D.-Y. 1998. Genetic programming with active data selection. In *Simulated Evolution and Learning: Second Asia-Pacific Conference on Simulated Evolution and Learning, SEAL'98. Selected Papers*, R. I. B. McKay, X. Yao, C. S. Newton, J.-H. Kim, and T. Furuhashi, Eds. LNAI, vol. 1585. Springer-Verlag, Australian Defence Force Academy, Canberra, Australia, 146–153. published in 1999.

Zhang, B.-T. and Joung, J.-G. 1999. Genetic programming with incremental data inheritance. 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. Vol. 2. Morgan Kaufmann, Orlando, Florida, USA, 1217–1224.

Zhang, B.-T. and Mühlenbein, H. 1995. Balancing accuracy and parsimony in genetic programming. *Evolutionary Computation 3,* 1, 17–38.

Zhang, B.-T. and Veenker, G. 1991. Focused incremental learning for improvedgeneralization with reduced training sets. In *Artificial Neural Networks*, T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, Eds. Elsevier, Amsterdam, Espoo, Finland, 227–232. Proc. ICANN-91.

Zitzler, E. and Thiele, L. 1999. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto evolutionary algorithm. *IEEE Transactions on Evolutionary Computation 3,* 4, 257–271.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and da Fonseca, V. G. 2003. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation 7,* 2, 117–132.