Geometric Semantic Genetic Programming and its Real-Life Applications

NOVA IMS
Information Management School

Leonardo Vanneschi
NOVA IMS
Universidade Nova De Lisboa
Portugal
lvanneschi@novaims.unl.pt

Leonardo Vanneschi Biographical Sketch

- 1996: University Degree in Computer Science at the University of Pisa (Italy)
- 1996-1999: Software Engineer in Industry
- 1999-2004: PhD Student and Assistant at the University of Lausanne (Switzerland)
  - Complex Systems, Machine Learning, Genetic Programming
- 2004-2011 : Assistant Professor at the Computer Science Department University of Milano-Bicocca, Italy
- 2011-2014 : Assistant Professor at the NOVA IMS, University Nova of Lisbon, Portugal
- 2014-... : Associate Professor with Tenure at the NOVA IMS, University Nova of Lisbon, Portugal

Objective

- Presenting new operators for Genetic Programming (geometric semantic operators)
- Their theoretical advantages
- Their practical (real-world) applications

Agenda

- Optimization Problems and Fitness Landscapes
- Genetic Algorithms (*geometric operators*)
- Genetic Programming (*geometric semantic operators*)
- Implementation of geometric semantic operators
- Real-Life Applications:
  - Drug Discovery (prediction of pharmacokinetic parameters)
  - Prediction of the Unified Parkinson's Disease Rating Scale Assessment
  - Prediction of high performance concrete strength
  - Prediction of the Relative Position of Computer Tomography (CT) Slices
  - Forecasting of Energy Consumption
- Discussion/Open Issues

Part I – Geometric Semantic GP

- Optimization Problems and Fitness Landscapes
- Genetic Algorithms (*geometric operators*)
- Genetic Programming (*geometric semantic operators*)
- Implementation of geometric semantic operators

Part II – Real-Life Applications

- Drug Discovery (prediction of pharmacokinetic parameters)
- Pred. of the Unif. Parkinson's Disease Rat. Scale Assess.
- Prediction of high performance concrete strength
- Pred. of the Rel. Pos. of Computer Tomography (CT) Slices
- Forecasting of Energy Consumption

Part III – Conclusions

- Discussion/Open Issues

PART I

Geometric Semantic Genetic Programming

Main reference for this first part:

L. Vanneschi.
An Introduction to Geometric Semantic Genetic Programming.
NEO 2015: Results of the Numerical and Evolutionary Optimization Workshop NEO 2015, held at September 23-25 2015 in Tijuana, Mexico. Pages 3-42.
O. Schutze, L. Trujillo, P. Lagrand and Y. Maldonado Editors.
Springer International Publishing.
2017.

isbn=978-3-319-44003-3
doi=10.1007/978-3-319-440033_1,
url=http://dx.doi.org/10.1007/978-3-319-44003-3_1

Let’s start with a very light introduction to optimization problem...

The content of this very first part is well-known to the audience, but it is used to fix a terminology that will be used in the continuation...

Optimization Problems

Informally

solving an optimization problem means to find the best solution(s) in a (typically huge) set of other candidate solutions

A little bit more formally

A pair: (S, f)

where S is the set of all possible solutions (we will call it search space) and f is a function:

$$f = S \rightarrow \mathbb{R}$$

f quantifies the quality of the solutions in S and it is called cost function or fitness function.

Optimization Problems - Definitions

An optimization problem is a *minimization problem* if it consists in looking for a solution $o \in S$, such that:

$$f(o) \leq f(i), \quad \forall i \in S$$

An optimization problem is a *maximization problem* if it consists in looking for a solution $o \in S$, such that:

$$f(o) \geq f(i), \quad \forall i \in S$$

Such a solution is called *global optimum* (minimum or maximum)

Hill Climbing

The most natural and immediate method to solve an optimization problem.

It consists in trying to improve fitness step by step (stepwise improvement) by means of the concept of **neighborhood**.

INITIALIZE(i_{start});

i := i_{start};

repeat

GENERATE(j from N_i);

if f(j) better than f(i) **then** i := j;

until f(i) better than f(j), $\forall$ j $\in$ N_i;

N_i = neighborhood of solution i

Hill Climbing - Example

Consider the following maximization problem:

$S = \{ i \mid \quad i \in \mathbb{N} \quad \& \quad 0 \leq i \leq 15 \}$

$\forall i \in S, \quad f(i) =$ number of "1"s in the binary representation of i

Neighborhood: $\quad j \in N_i \Leftrightarrow |j - i| = 1$

Steps of the algorithm:

- Current solution (randomly generated): $i =$ ***6 (0110)*** $f(i) = 2$
  neighbors of $i =$ ***5 (0101)***, ***7 (0111)*** $f(5) = 2$, $f(7) = 3$ new current solution: $i := 7$
- neighbors of $i =$ ***6 (0110)***, ***8 (1000)*** $f(6) = 2$, $f(8) = 1$ algorithm terminates

Solution returned: $i =$ ***7 (0111)*** $f(i) = 3$
Global optimum: $o =$ ***15 (1111)*** $f(o) = 4$

The solution returned by the Hill Climbing is a "local optimum".

Local Optima

A solution $j \in S$ is called local optimum (as regards a neighborhood structure N), if:

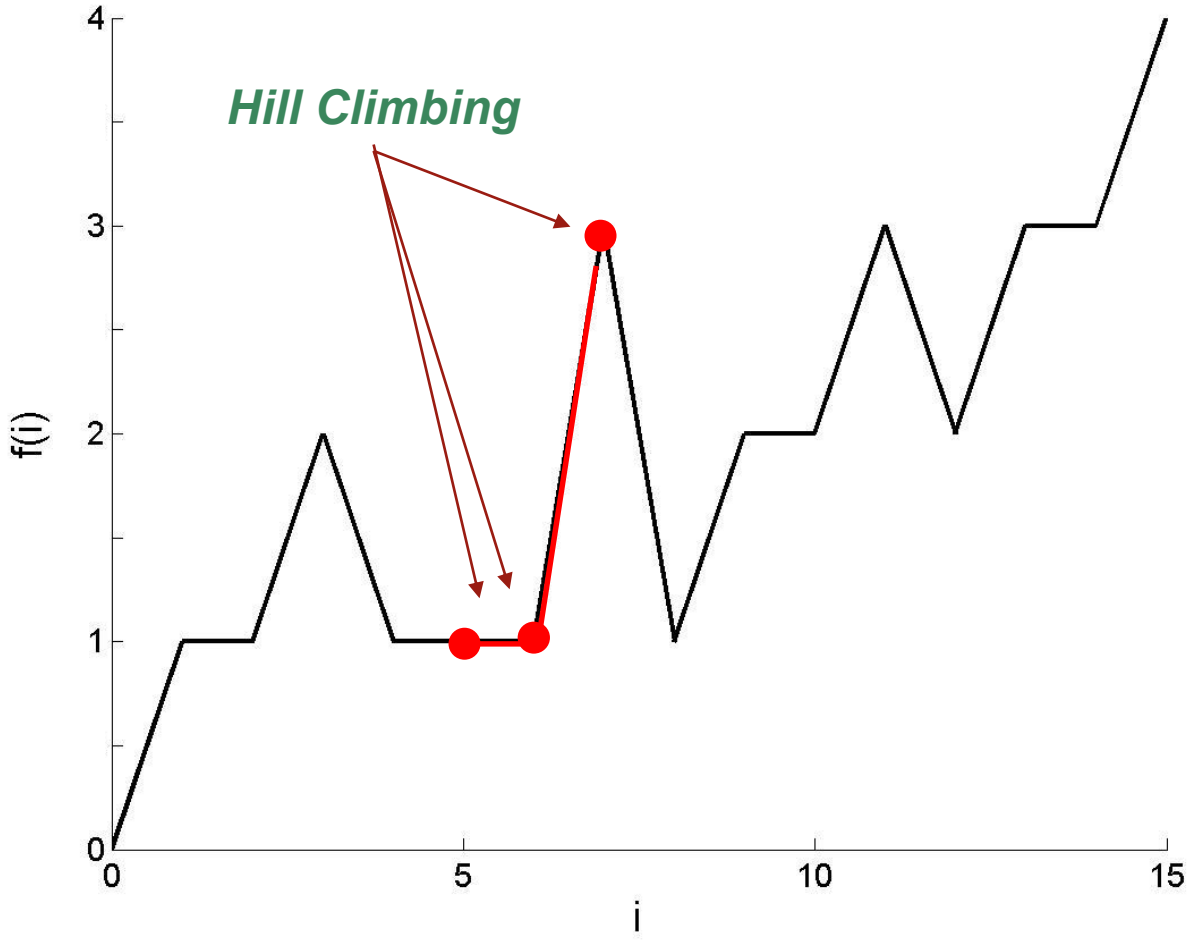
for minimization problems:
$f(j) \leq f(i) \quad \forall i \in N_j$

for maximization problems:
$f(j) \geq f(i) \quad \forall i \in N_j$

Hill Climbing always returns a local optimum (not necessarily the global one).

Fitness Landscape

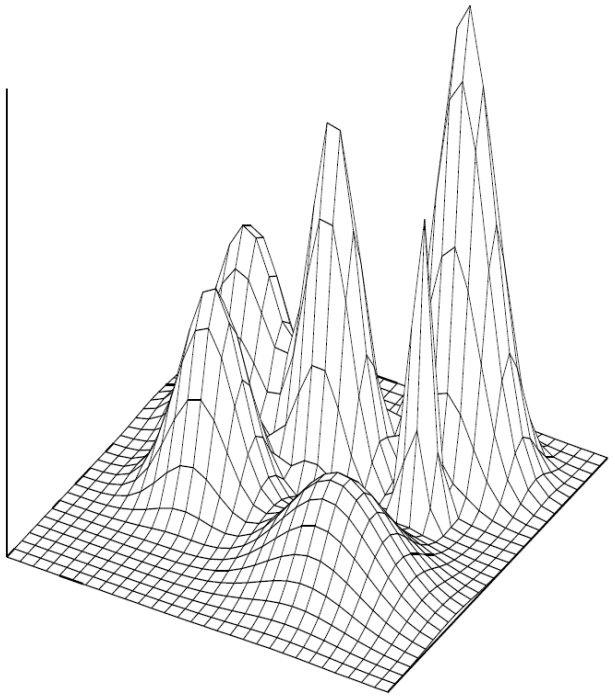
Plot: horizontal axis: solutions in the search space (ordered according to the neighborhood structure); vertical axis: fitness.



Fitness Landscape

Fitness landscape $(\mathcal{S}, \mathcal{V}, f)$:

- $\mathcal{S}$: set of potential solutions,
- $\mathcal{V} : \mathcal{S} \rightarrow 2^{\mathcal{S}}$: neighborhood function,
- $f : \mathcal{S} \rightarrow \mathbb{R}$: fitness function.



$\mathcal{V} : \mathcal{S} \rightarrow 2^{\mathcal{S}}$: neighborhood function

$\forall x \in \mathcal{S}$,

$$\mathcal{V}(x) = \{y \in \mathcal{S} \mid y = op(x)\}$$

$$\mathcal{V}(x) = \{y \in \mathcal{S} \mid d(y,x) \leq 1\}$$

(courtesy of Sebastien Verel)

Importance of Fitness Landscape

It gives a *visual intuition* of the *facility* or *difficulty* of a search agent (like Hill Climbing, but also Evolutionary Algorithms) ***to find the global optimum***.
For instance:

- Smooth landscape, with only one "peak" (global optimum)
  easy problem
- Rugged landscape, with many local optima
  hard problem

Limitation of fitness landscapes

It is generally ***impossible to draw*** a fitness landscape:

- Huge search space
- Huge neighborhoods *(multi-dimensionality!)*

Remark that...

if we consider exactly the same problem, but with a ***different neighborhood structure***, the fitness landscape changes and Hill Climbing easily finds easily the global optimum:

$$S = \{ i \mid \quad i \in \mathbb{N} \quad \& \quad 0 \leq i \leq 15 \}$$

$\forall i \in S, \quad f(i) =$ number of "1"s in the binary representation of i

Neighborhood: $j \in N_i \Leftrightarrow$ the binary repr. of j and i differ by just 1 bit

There are ***no local optima*** in this fitness landscape!
Unimodal fitness landscape.
(every individual that is different from the global optimum has at least one neighbor better than him, that can be obtained by changing a 0 into a 1).

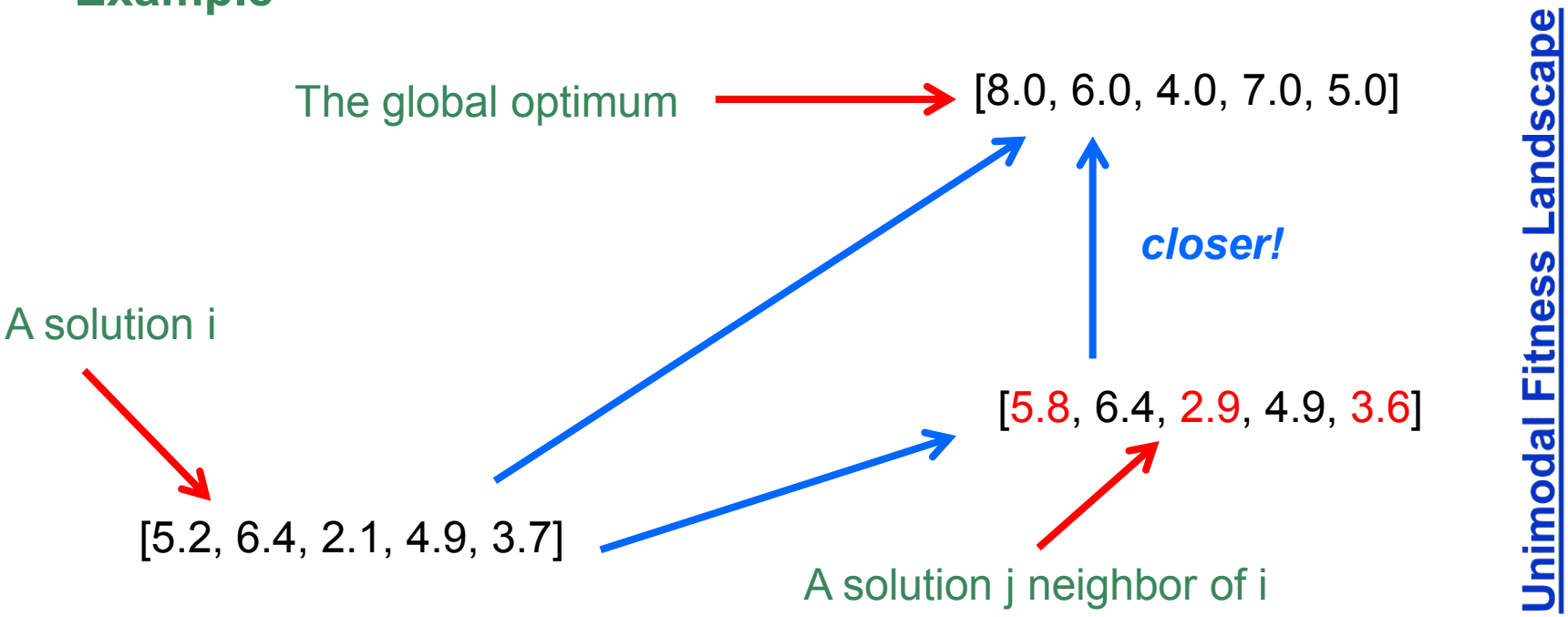
Another Case

S = *{ vectors of prefixed length of real numbers included in [0,10] }*

$\forall i \in S$, $f(i)$ = distance to a prefixed (and known and unique) global optimum

Neighborhood: $j \in N_i \Leftrightarrow$ *j is equal to i except for the random perturbation of some of its coordinates of a quantity included in [0,1].*

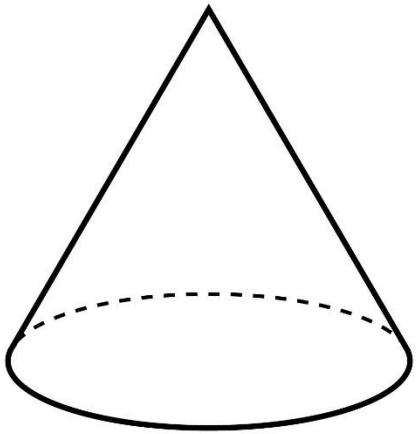
Example



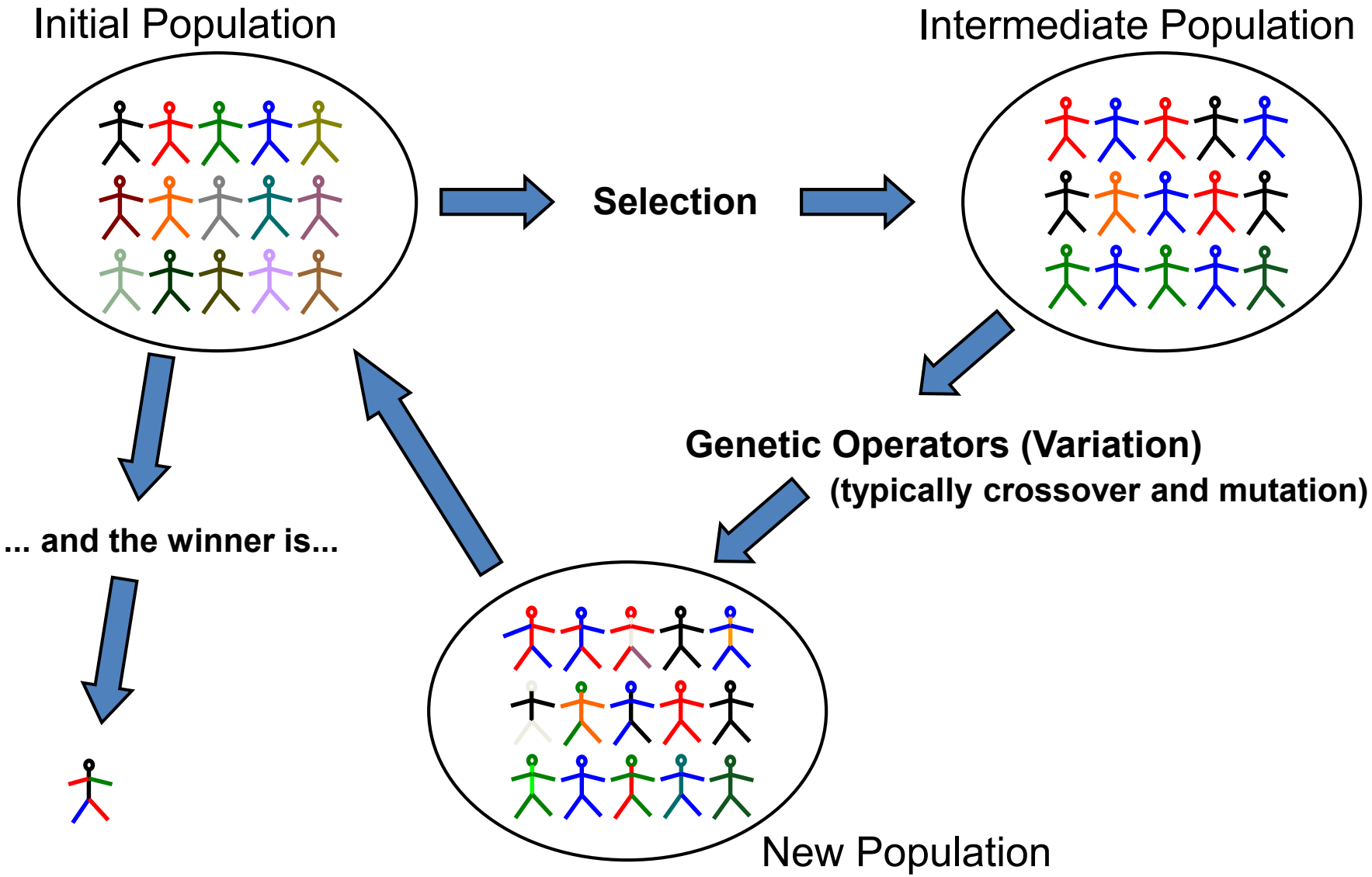
Terminology about the previous example

The operator that is “related” to the neighborhood structure of the previous example is called ***ball mutation***.

Let us give a name to the previous problem:
Continuous **O**ptimization with **N**otorious **O**ptimum (**CONO**)



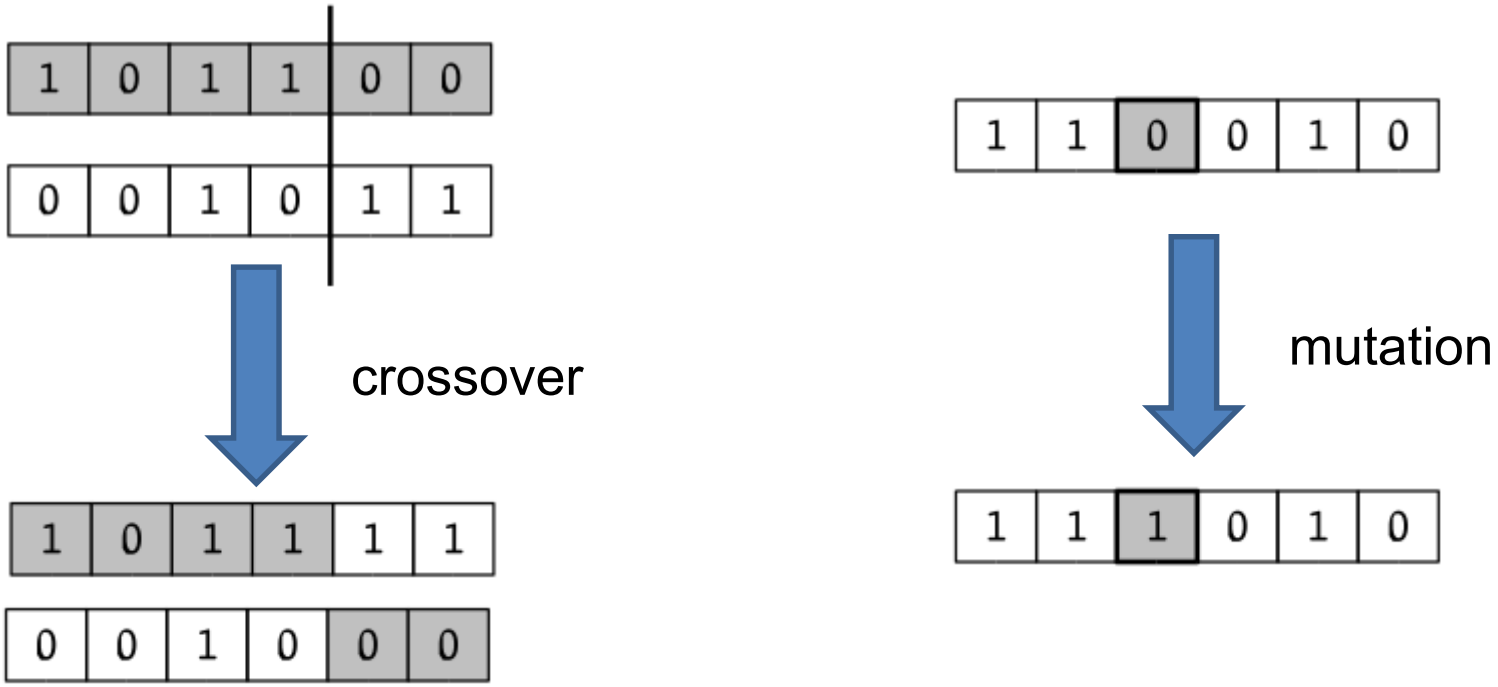
Evolutionary Algorithms



Genetic Algorithms (GAs)

Solutions/Individuals = *Strings of prefixed length*

"Traditional situation": the values in each allele are discrete and:



Genetic Algorithms (GAs)

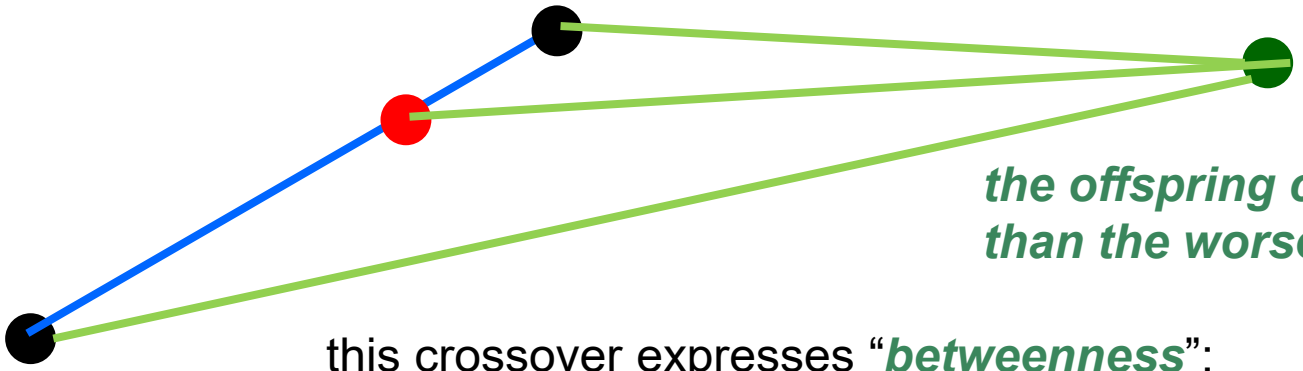
... but GAs can work also on vectors of *continuous* values.

In that case, many operators have been introduced.

Interesting: **Geometric operators** [Moraglio and Poli, 2006]:

Geometric Crossover

Coordinates of the (unique) offspring are the weighted average of the corresponding coordinates of the parents (with weighs in [0,1] whose sum is 1).



the offspring cannot be worse than the worse of the parents

this crossover expresses "***betweenness***":
the offsrping stands between (in the line joining) the parents

Genetic Algorithms (GAs)

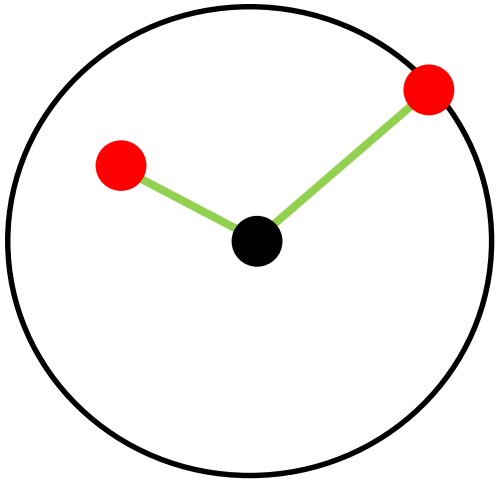
... but GAs can work also on vectors of continuous values.

In that case, many operators have been introduced.

Interesting: **Geometric operators** [Moraglio and Poli, 2006]:

Geometric Mutation

Ball mutation



Unimodal fitness landscape on the CONO problem.

Genetic Programming (GP)

An evolutionary algorithm in which solutions/individuals are ***computer programs***.

Typically ("Lisp-like") trees.

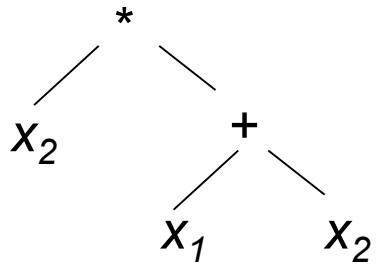
Example (Symbolic Regression)

Given the set of data:

input target

$$H = \left(\begin{array}{cc|c} 2 & 4 & 10 \\ 3 & 5 & 13 \end{array}\right)$$

A possible individual is:



It represents the program/function/expression: $P(x_1, x_2) = x_2 * (x_1 + x_2)$

And one possible fitness could be:

$fitness(P) = |P(2,4) - 10| + |P(3,5) - 13| =$

$|4 * (2+4) - 10| + |5 * (3+5) - 13| = |24 - 10| + |40 - 13| = 14 + 27 = 41$

GP as a Machine Learning Method

- **Known**: the correct outputs for a fixed given set of inputs $\{I_i, O_i\}$
- **Sought**: a function belonging to a certain class that interpolates those points, i.e., $f(I_i) = O_i$ for any i
- **Output vector**: the vector of the outputs of f is $f(I)=(f(I_i))$
- **Fitness**: a measure on the error on the training set, i.e.,***distance*** between the output vectors of f and the target output vector $F(f)=D(f(I),O)$ (ERROR AS DISTANCE)

We are talking of ***Supervised Learning***.

What is «Semantics» ?

[Nguyen at al., 2011], [Moraglio et al., 2012], …

Many definition exist, but in GP the most used one is:

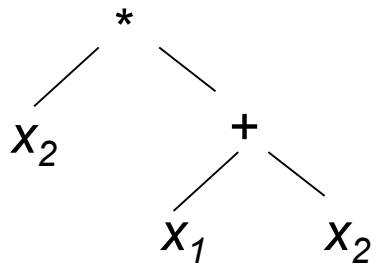
The vector of outputs of a program on the different training data

Example

Given the set of data:

$$H = \left(\begin{array}{cc|c} 2 & 4 & 10 \\ 3 & 5 & 13 \end{array}\right)$$

The individual:



That represents function: $P(x_1, x_2) = x_2 * (x_1 + x_2)$

has a semantics qual to: $[P(2,4), P(3,5)] = \mathbf{[24, 40]}$

Semantics

Let $\mathbf{X} = [\vec{x}_1, \vec{x}_2, ..., \vec{x}_n]$ be the set of input data (fitness cases) of a *supervised learning* problem and $\vec{t} = [t_1, t_2, ..., t_n]$ the vector of the respective expected output or target values.

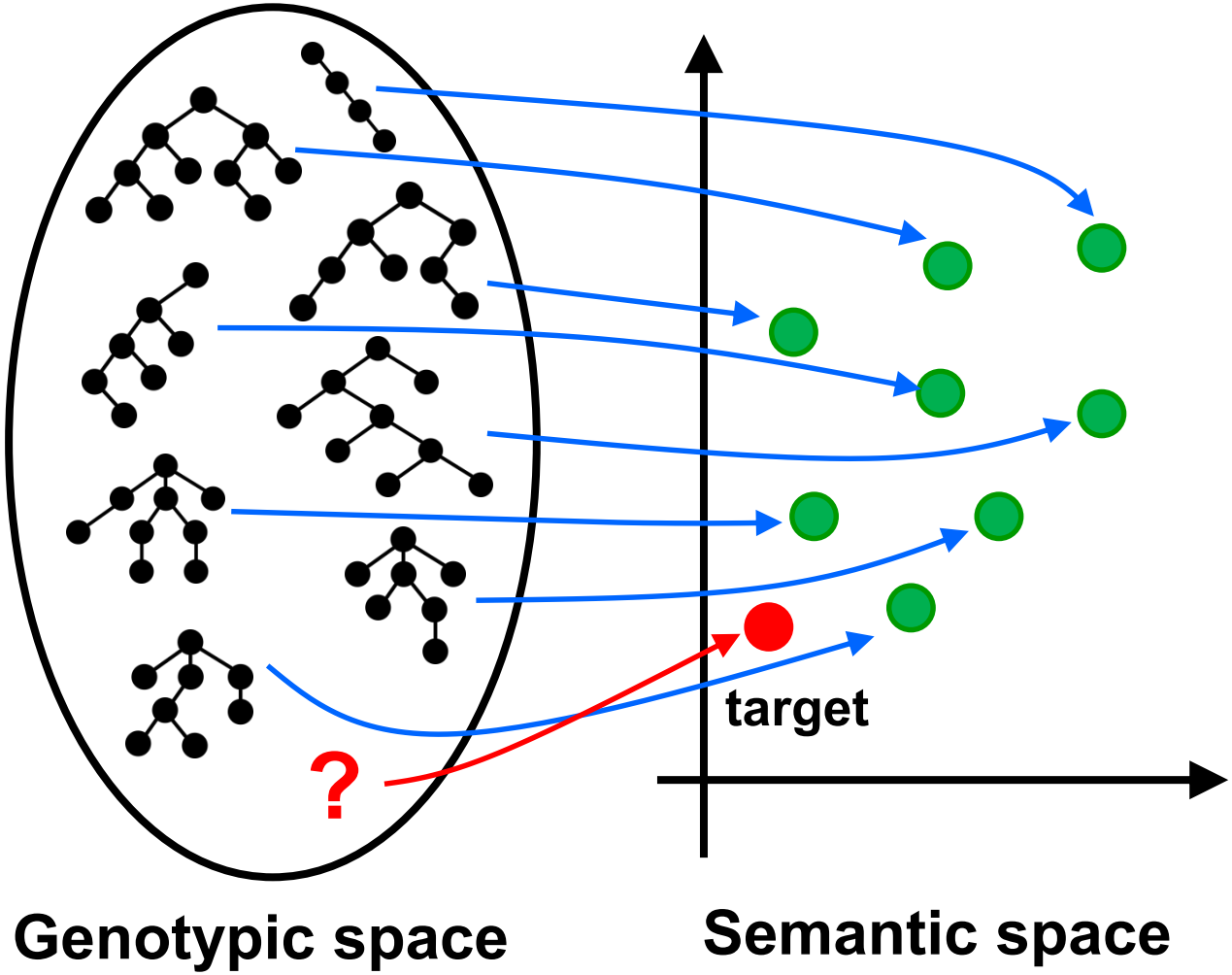
A GP individual (or program) P can be seen as a function that, for each input vector $\vec{x}_i$ returns the scalar value $P(\vec{x}_i)$.

We define semantics of an individual P the vector:

$$\vec{s}_P = [P(\vec{x}_1), P(\vec{x}_2), ..., P(\vec{x}_n)]$$

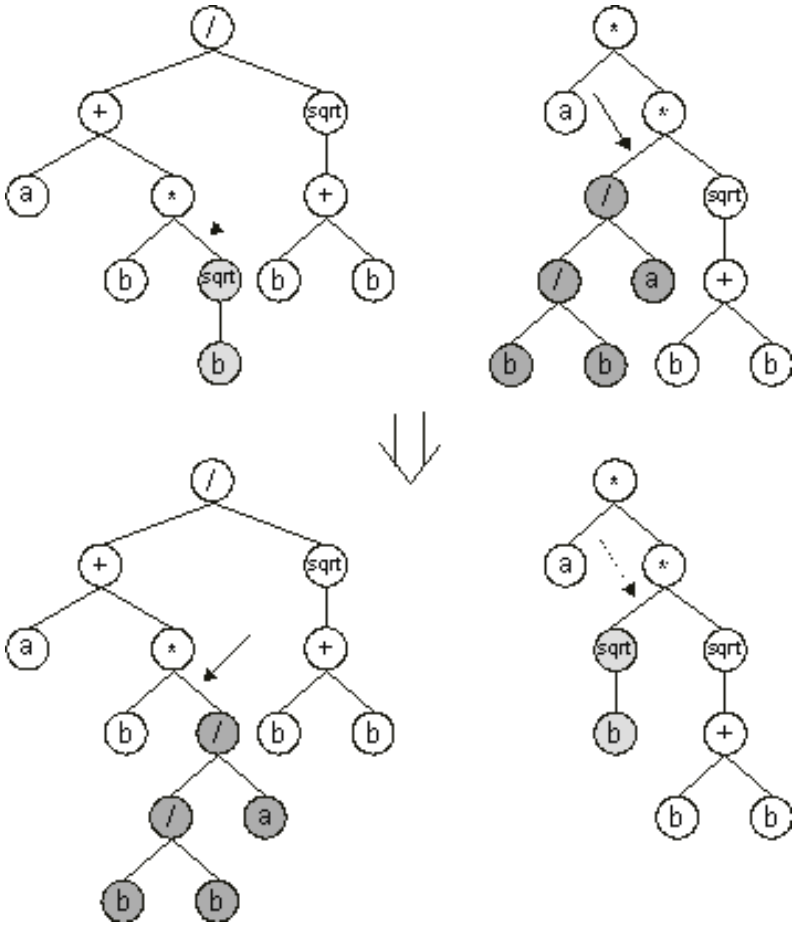
This is a ***point*** in a ***n***-dimensional space

Semantic Space

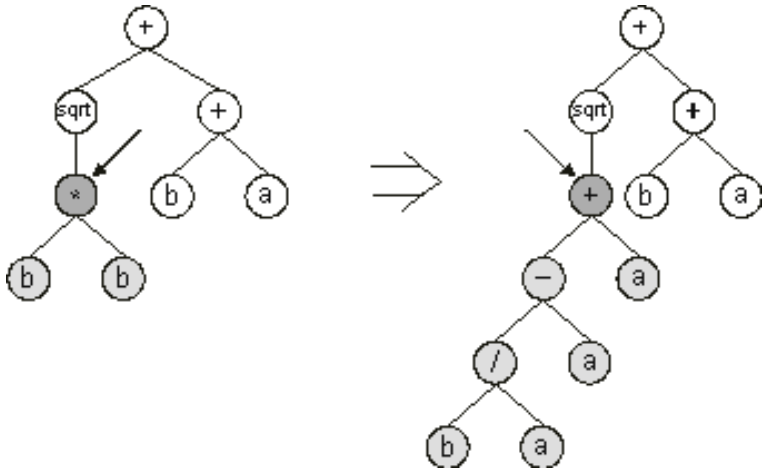


The ***target*** is also represented by a ***point*** in the semantic space and usually it does not correspond to the origin

«Traditional» GP operators



crossover



mutation

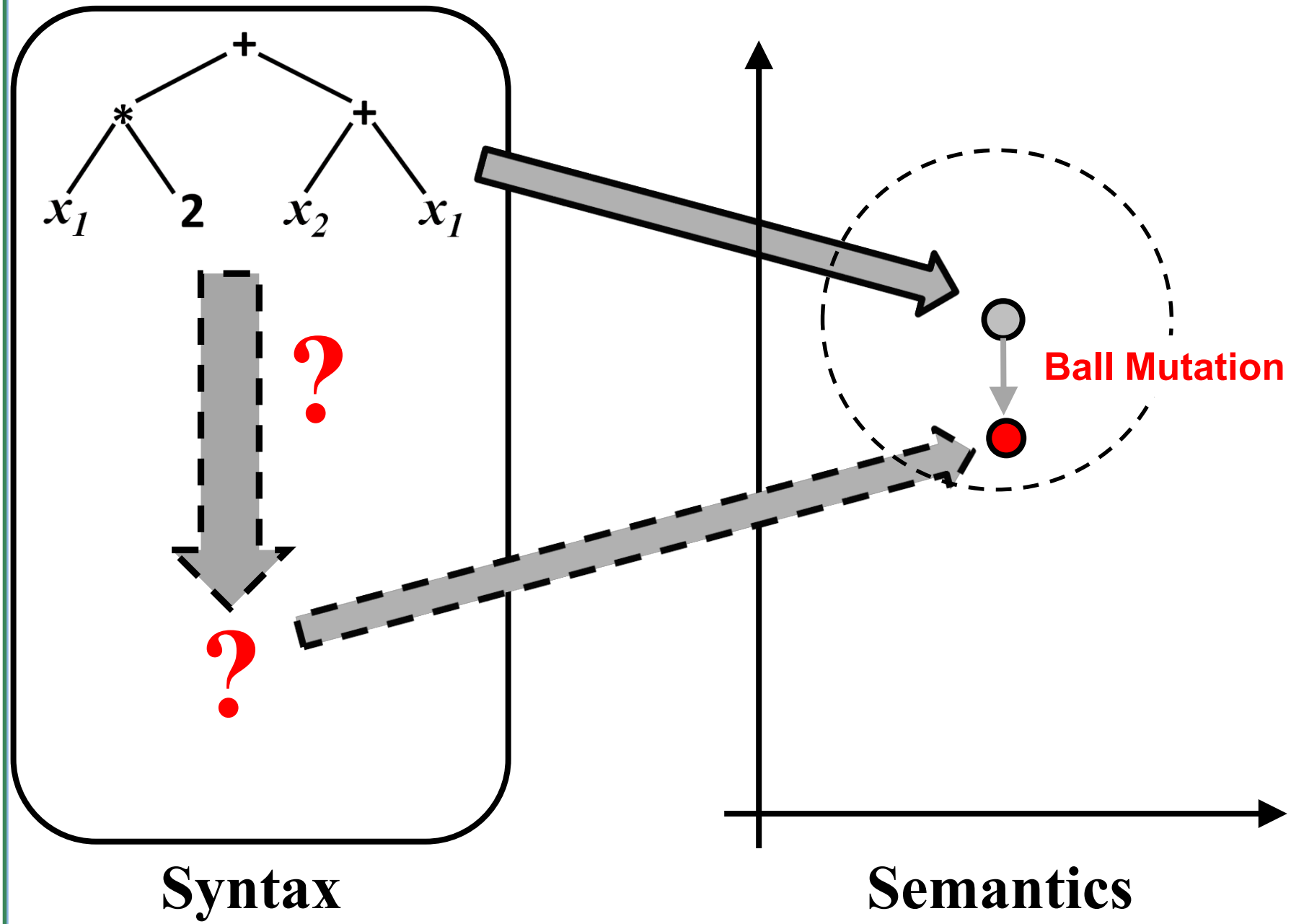
«Traditional» GP operators

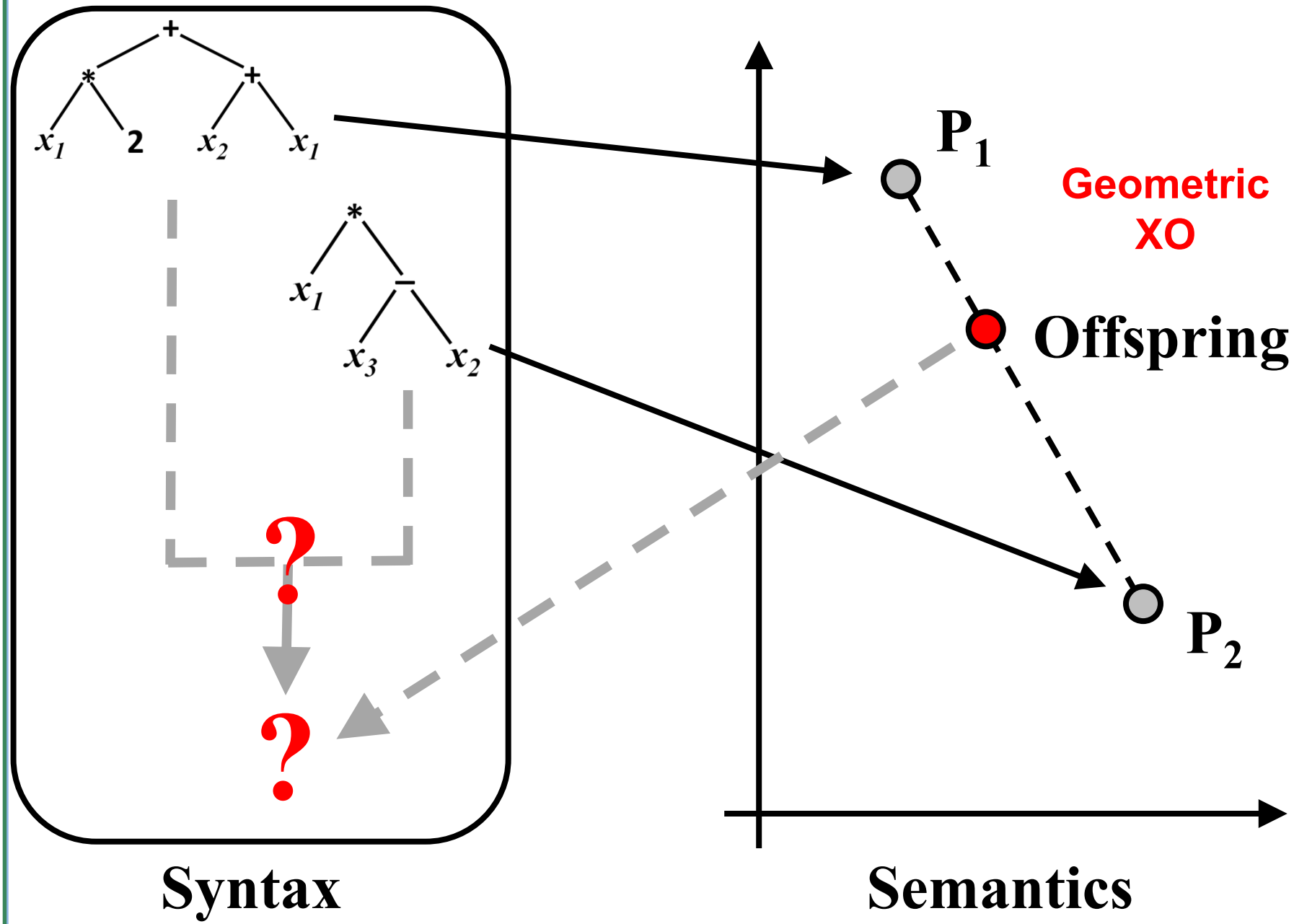
… Produce offspring by blind syntactic manipulation of parent parse trees, regardless of their semantics.

… Preserve syntactic “genetic material”, but what is their effect on semantics?

Objective

Is it possible to define transformations on the syntax of individuals that have known effects on their semantics?


+
*
+
x_1
2
x_2
x_1
?
?
Ball Mutation
Syntax
Semantics


+
*
+
x1
2
x2
x1
*
x1
–
x3
x2
?
?
P1
Geometric
XO
Offspring
P2
Syntax
Semantics

Impact

Let us assume that we are able to find a transformation on the syntax of an individual whose effect is ***ball mutation on the semantic space***.

This transformation, if known, would induce a ***unimodal fitness landscape*** on ***every problem*** consisting in matching input data into known targets (e.g. regressions and classifications),

GP should have a good evolvability on those problems, at least on training data (we are ***mapping the problem into the CONO***).

The same also holds for transformations on pairs of solutions that correspond to GA semantic crossovers.

Is it a dream?

Yes... but turning into reality

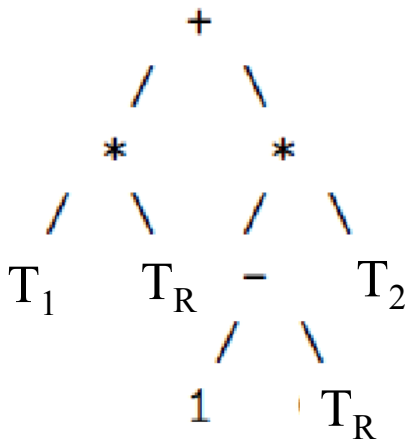
Even though with a big drawback (discussed later) those operators have been defined:

A. Moraglio, K. Krawiec, and C. G. Johnson.
Geometric semantic genetic programming.
In C. A. Coello Coello, et al., editors, *Parallel Problem Solving from Nature, PPSN XII* (part 1), volume 7491 of Lecture Notes in Computer Science, pages 21–31. Springer, 2012.

Geometric Semantic Crossover [Moraglio et al., 2012]

Definition 1. (Geometric Semantic Crossover). *Given two parent functions* $T_1, T_2 : \mathbb{R}^n \rightarrow \mathbb{R}$, *the geometric semantic crossover returns the real function* $T_{XO} = (T_1 \cdot T_R) + ((1 - T_R) \cdot T_2)$, *where* T_R *is a random real function whose output values range in the interval* $[0, 1]$.

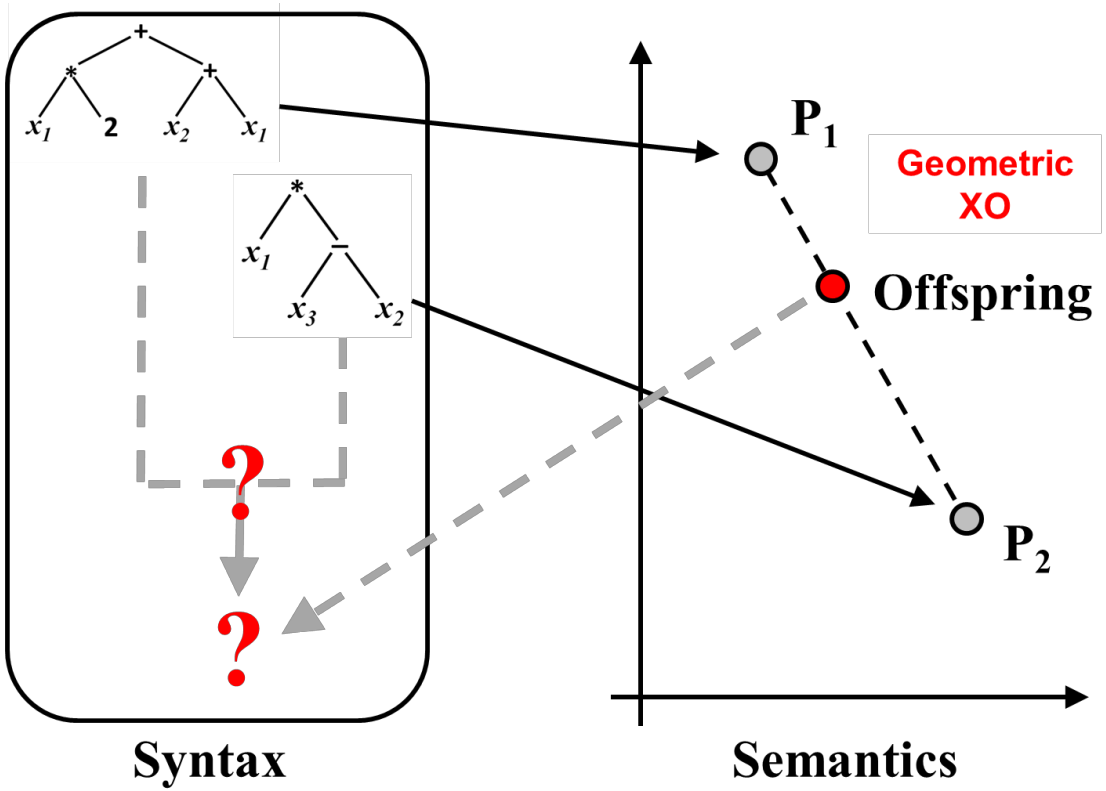
T_{XO} =



T_R = Random function with codomain [0, 1]

Geometric Semantic Crossover [Moraglio et al., 2012]

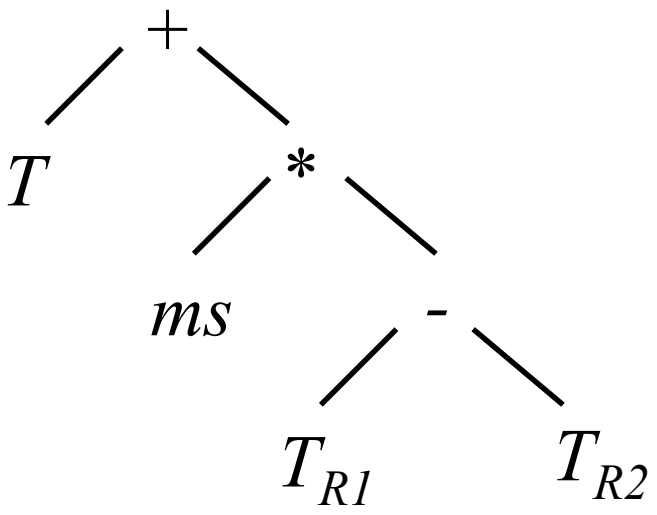
The (only) offspring generated by this ***crossover has a semantic vector*** that is a ***linear combination of the semantics*** of the parents with random coefficients included in [0,1] and whose sum is equal to 1.



Geometric Semantic Mutation [Moraglio et al., 2012]

Definition 2. (Geometric Semantic Mutation). *Given a parent function* $T : \mathbb{R}^n \rightarrow \mathbb{R}$, *the geometric semantic mutation with mutation step* ms *returns the real function* $T_M = T + ms \cdot (T_{R1} - T_{R2})$, *where* T_{R1} *and* T_{R2} *are random real functions.*

T_M =



T_{R1} , T_{R2} = Random functions

Geometric Semantic Mutation [Moraglio et al., 2012]

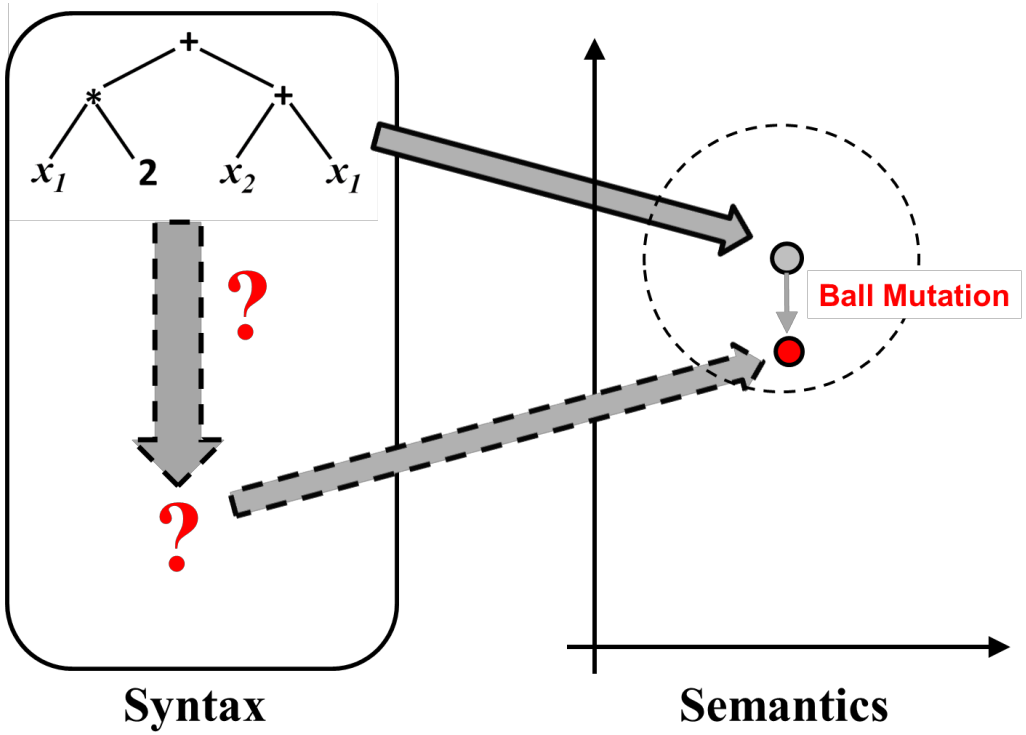
Each element of the semantic vector of the offspring is a *"weak" perturbation* of the corresponding element in the parent's semantics.

"Weak" because $(T_{R1} - T_{R2})$ is *centered in zero*

It's *importance* can be tuned by *ms*.

Geometric Semantic Mutation [Moraglio et al., 2012]

It corresponds to *ball mutation in the semantic space*, so the fitness landscape it induces is unimodal (we match each problem consisting in matching inputs into targets into the CONO problem!).



Cool !! But....

Geometric Semantic Operators

Moraglio et al. show interesting results on a set of benchmarks, but....

XO(T1, T2) = (T1 * TR) + ((1 – TR) * T2)

At generation 2 (if we use only crossover), all the trees have this shape, so to create the next generation:

XO((T1 * TR1) + ((1 – TR1) * T2), (T3 * TR2) + ((1 – TR2) * T4)) =

(((T1 * TR1) + ((1 – TR1) * T2)) * TR3) + ((1 – TR3) * ((T3 * TR2) + ((1 – TR2) * T4)))

Now assume to take two trees of this shape and apply the crossover to generate the offspring of generation 3

And assume to iterate this for hundreds of generations!

Drawback of Geometric Semantic Operators

These operators, by construction, always produce offspring that are larger than their parents, causing a ***fast growth*** in the size of the individuals (proven in [Moraglio et al., 2012]

This renders them ***useless in practice***.

A solution that has been proposed: "***simplification***" of the individuals during the evolution. But….

Our Contribution

In:

A New Implementation of Geometric Semantic GP Applied to Predicting Pharmacokinetic Parameters.
L. Vanneschi, M. Castelli, L. Manzoni, S. Silva.
Accepted for publication in the *EuroGP 2013* Proceedings Lecture Notes in Computer Science.

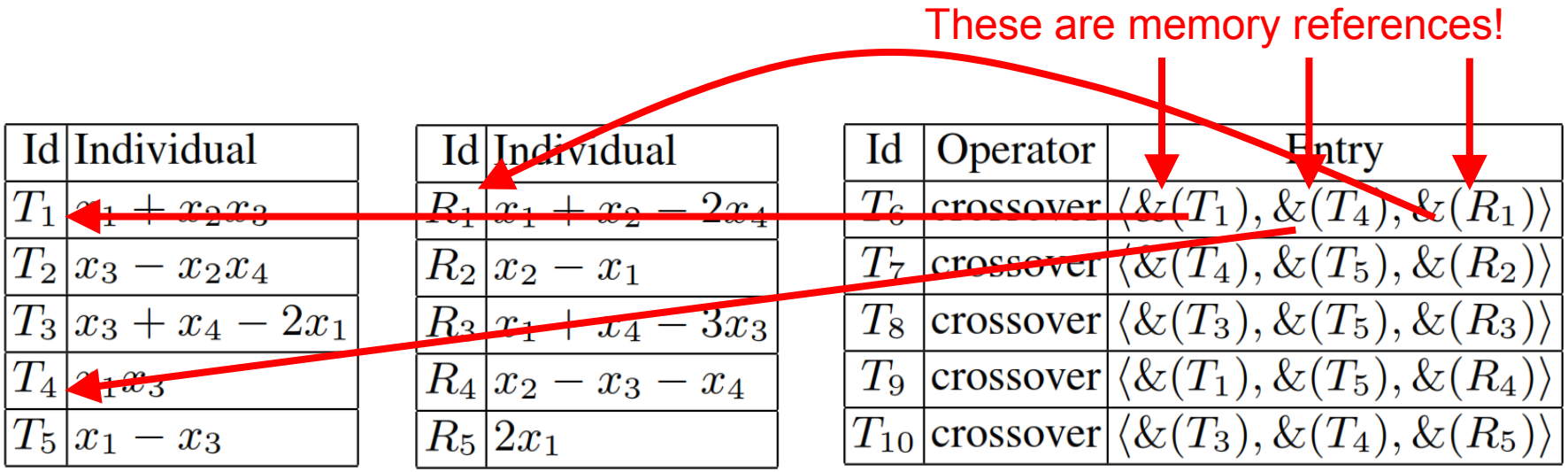
We propose a ***new implementation*** of Moraglio's geometric semantic operators that is ***efficient*** and thus allows us to use them (for the first time) on complex real-life applications!

The New Implementation

We create the initial population as in standard GP and we store the trees.

We create all the random trees that we need to produce the next population.

We store the next population like this.



Id	Individual
T_1	$x_1 + x_2 x_3$
T_2	$x_3 - x_2 x_4$
T_3	$x_3 + x_4 - 2x_1$
T_4	$x_1 x_3$
T_5	$x_1 - x_3$

Id	Individual
R_1	$x_1 + x_2 - 2x_4$
R_2	$x_2 - x_1$
R_3	$x_1 + x_4 - 3x_3$
R_4	$x_2 - x_3 - x_4$
R_5	$2x_1$

Id	Operator	Entry
T_6	crossover	$\langle \&(T_1), \&(T_4), \&(R_1) \rangle$
T_7	crossover	$\langle \&(T_4), \&(T_5), \&(R_2) \rangle$
T_8	crossover	$\langle \&(T_3), \&(T_5), \&(R_3) \rangle$
T_9	crossover	$\langle \&(T_1), \&(T_5), \&(R_4) \rangle$
T_{10}	crossover	$\langle \&(T_3), \&(T_4), \&(R_5) \rangle$

Id	Individual				
T_1	$x_1 + x_2x_3$	7.34	8.62	9.51	4.07
T_2	$x_3 - x_2x_4$	9.73	4.29	5.26	1.45
T_3	$x_3 + x_4 - 2x_1$	2.92	3.76	5.23	6.24
T_4	x_1x_3	7.28	1.78	3.26	5.74
T_5	$x_1 - x_3$	2.57	4.67	3.22	6.91

Id	Individual				
R_1	$x_1 + x_2 - 2x_4$	2.64	3.28	5.93	4.29
R_2	$x_2 - x_1$	6.94	7.53	8.53	2.65
R_3	$x_1 + x_4 - 3x_3$	4.84	3.56	2.76	9.76
R_4	$x_2 - x_3 - x_4$	4.37	5.94	2.59	1.85
R_5	$2x_1$	4.67	3.27	2.57	7.47

Id	Operator	Entry				
T_6	crossover	$\langle \&(T_1), \&(T_4), \&(R_1) \rangle$	.....	.....	.....	.....
T_7	crossover	$\langle \&(T_4), \&(T_5), \&(R_2) \rangle$	.....	.....	.....	.....
T_8	crossover	$\langle \&(T_3), \&(T_5), \&(R_3) \rangle$	.....	.....	.....	.....
T_9	crossover	$\langle \&(T_1), \&(T_5), \&(R_4) \rangle$	.....	.....	.....	.....
T_{10}	crossover	$\langle \&(T_3), \&(T_4), \&(R_5) \rangle$	.....	.....	.....	.....

Semantics

Storing also the semantics of each individual allows us to calculate the fitness without evaluating the whole expresson!!

Obtainable directly from here

Semantics in the next population

The New Implementation

Before passing to the next generation:

Id	Individual
T_1	$x_1 + x_2x_3$
~~T_2~~	~~$x_3 \quad x_2x_4$~~
T_3	$x_3 + x_4 - 2x_1$
T_4	x_1x_3
T_5	$x_1 - x_3$

Id	Individual
R_1	$x_1 + x_2 - 2x_4$
R_2	$x_2 - x_1$
R_3	$x_1 + x_4 - 3x_3$
R_4	$x_2 - x_3 - x_4$
R_5	$2x_1$

Id	Operator	Entry
T_6	crossover	$\langle \&(T_1), \&(T_4), \&(R_1) \rangle$
T_7	crossover	$\langle \&(T_4), \&(T_5), \&(R_2) \rangle$
T_8	crossover	$\langle \&(T_3), \&(T_5), \&(R_3) \rangle$
T_9	crossover	$\langle \&(T_1), \&(T_5), \&(R_4) \rangle$
T_{10}	crossover	$\langle \&(T_3), \&(T_4), \&(R_5) \rangle$

T_2 is not used anymore: it can be deleted from memory !

The New Implementation

For producing the next generation:

We create another pool of random trees to produce the next population.

We store the next population in another table of pointers.

We eventually simplify removing trees that will never be used).

Id	Individual
T_1	$x_1 + x_2 x_3$
T_2	$x_3 - x_2 x_4$
T_3	$x_3 + x_4 - 2x_1$
T_4	$x_1 x_3$
T_5	$x_1 - x_3$

Id	Individual
R_1	$x_1 + x_2 - 2x_4$
R_2	$x_2 - x_1$
R_3	$x_1 + x_4 - 3x_3$
R_4	$x_2 - x_3 - x_4$
R_5	$2x_1$
R_6	$x_3 * x1$
R_7	$x_4 + x_2 - x_1$
R_8	$x_3 - 7x_1 + x_4$
R_9	$x_1 + 2x_3$
R_{10}	$x_4 * x_2 - x_1 * x_3$

Id	Operator	Entry
T_6	crossover	$\langle \&(T_1), \&(T_4), \&(R_1) \rangle$
T_7	crossover	$\langle \&(T_4), \&(T_5), \&(R_2) \rangle$
T_8	crossover	$\langle \&(T_3), \&(T_5), \&(R_3) \rangle$
T_9	crossover	$\langle \&(T_1), \&(T_5), \&(R_4) \rangle$
T_{10}	crossover	$\langle \&(T_3), \&(T_4), \&(R_5) \rangle$
T_{11}	crossover	$\langle \&(T_3), \&(T_7), \&(R_6) \rangle$
T_{12}	crossover	$\langle \&(T_1), \&(T_2), \&(R_7) \rangle$
T_{13}	crossover	$\langle \&(T_8), \&(T_10), \&(R_8) \rangle$
T_{14}	crossover	$\langle \&(T_3), \&(T_8), \&(R_9) \rangle$
T_{15}	crossover	$\langle \&(T_4), \&(T_6), \&(R_{10}) \rangle$

The New Implementation – Comp. Complexity

We keep in memory:

- (a subset of) the initial population
- a pool of random trees
- a table of memory references

maintains a constant size (but can be simplified)

increases in size at each generation (but can be simplified)

increases in size at each generation (but can be simplified)

Cost of evolving a population of n individuals for g generations.

At every generation, we need O(n) space to store the new individuals. Thus, we need O(ng) space in total. Since we need to do only O(1) operations for any new individual (since the fitness can be computed using the fitness of the parents), the time complexity is **O(ng)**.
Thus, we have a ***linear*** complexity with respect to population size and number of generations.

Video

(a movie will be shown here)

Implementation of GSGP

Available for free at:

http://gsgp.sourceforge.net/

The last step: expression reconstruction

At the end of a run, we have to reconstruct the trees, and those trees are huge, but:

- Usually we just need ***one individual*** (the best on training), so we can do the reconstruction only once.
- We can do it ***offline***, at the end of the run, so that we do not slow down the evolution

PART II

Real-Life Applications

Prediction of Pharmacokinetic Parameters

L. Vanneschi. Improving genetic programming for the prediction of pharmacokinetic parameters. *Memetic Computing*, 6(4):255–262, 2014

L. Vanneschi, S. Silva, M. Castelli, and L. Manzoni. Geometric semantic genetic programming for real life applications. In R. Riolo, et al., editors, *Genetic Programming Theory and Practice XI*, Genetic and Evolutionary Computation, pages 191–209. Springer New York, 2014

Drug Discovery

Drug discovery is the process by which new candidate medications are discovered and commercialized

It is an *expensive*, *slow* and *risky* business.

A recent analysis suggests that it costs on average more than $1 billion to launch a potentially technically successful drug, and it takes on average 12.5 years.

(source: http://emedicine.medscape.com/article/169814-overview)

The case of Troglitazone

- Approved in 1997 as an antidiabetic drug, and seemed set to become a global “bestseller”
- In 3 years it caused more than 90 verified cases of hepatotoxicity (with consequent serious liver injuries), among which 63 liver-failure deaths
- Withdrawn from the market in 2000 by the Food and Drug Administration (FDA)
- It is estimated that this withdrawal costed approximately $136 million

(sources: http://emedicine.medscape.com/article/169814-overview and http://84.19.28.94/troglitazone-story)

The case of Troglitazone is not isolated

Since 2000, more than 900 drugs have been reported to cause liver injury, and drugs account for approximately 40% of all instances of fulminant hepatic failure.

Why do this happens?

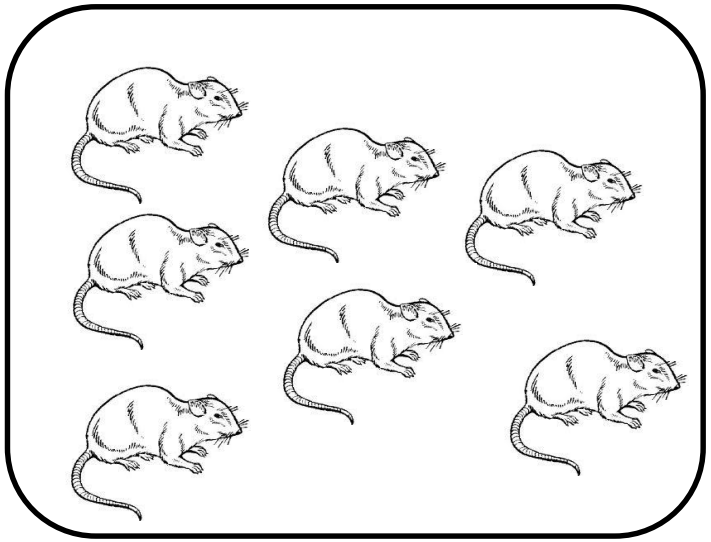
Because predicting drugs toxity is:

- Difficult
- (Very!) Expansive

(source: http://emedicine.medscape.com/article/169814-overview)

How the Toxicity of a Potential New Drug is Measured

test animals



Feed all of them slowly increasing amounts of drug

Until 50% of them die

The total amount of drug fed is called Median Oral Lethal Dose (**LD50**) and it is nowadays one of the most accepted measures of toxicity

Problems with Toxicity Measurements

The sample of cavies has to be statistically significant and there is an extremely large variance in the amount of drug needed to kill a test animal, so many cavies are needed in order to obtain significant results.

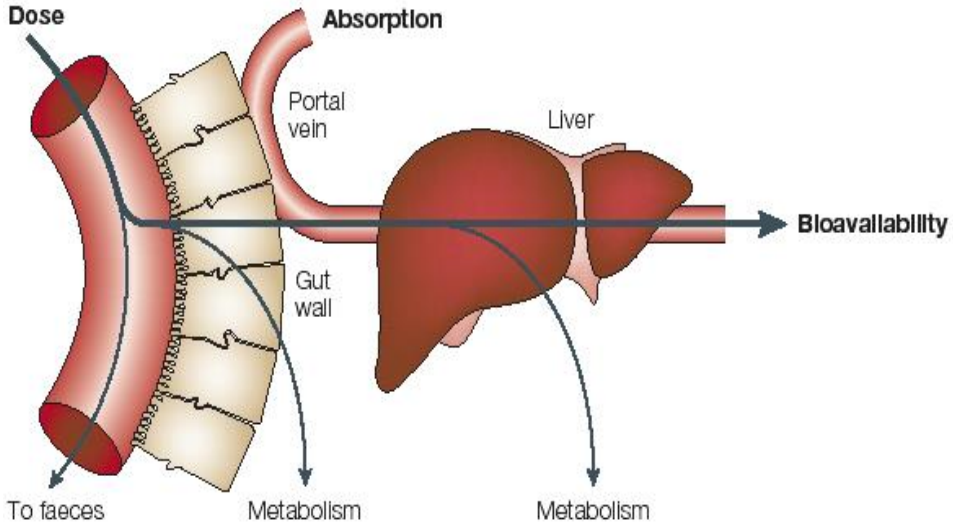
Furthermore…

Toxicity is only one important (pharmacokinetic) parameter that has to be known before commercializing a drug!

In this presentation:

- Human oral bioavailability
- Plasma protein binding level

Oral Bioavailability (%F)



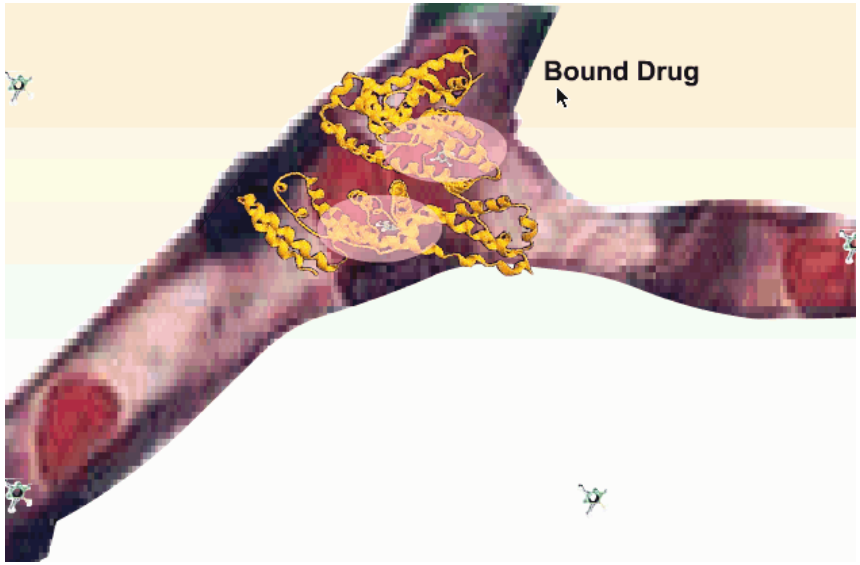
- Drugs have to be absorbed from the gut wall and to enter into systemic circulation in the portal vein.
- Carried by the blood flux, molecules arrive in the liver, where there are some biochemical processes that try to demolish them.

Oral Bioavailability = The percentage of molecules initially submitted that exit the liver and enter the blood circulation.

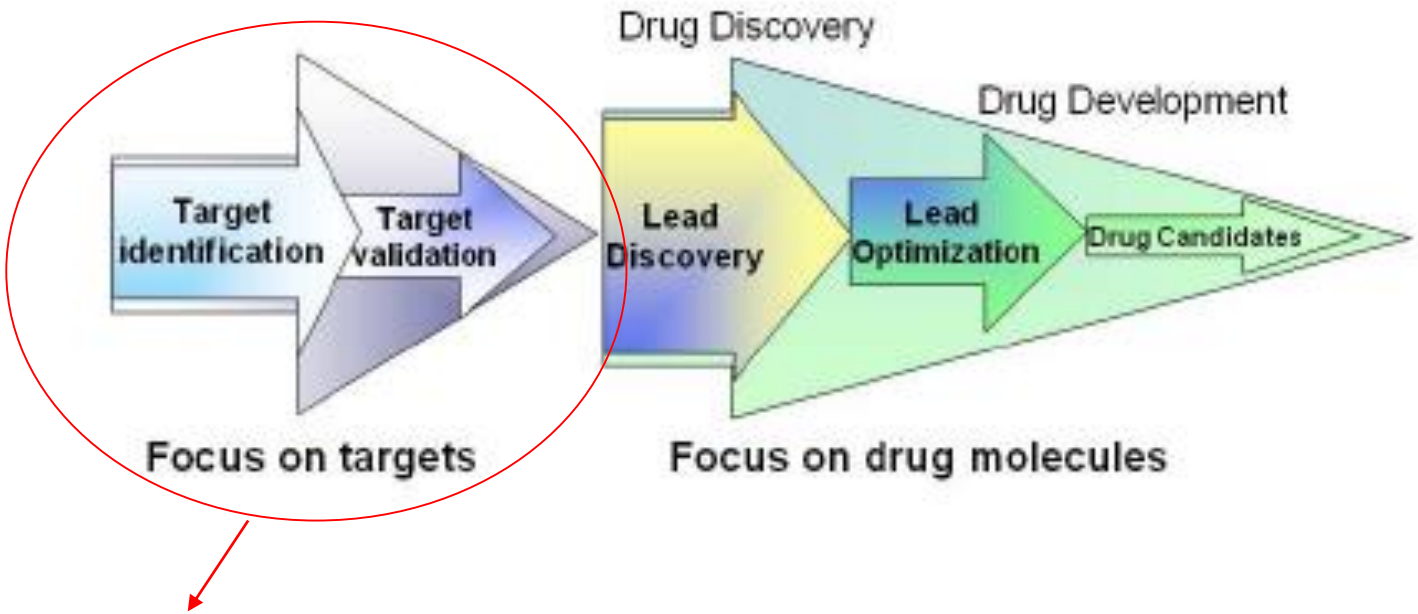
Plasma Protein Binding Levels (%PPB)

The percentage of the initial drug dose which binds plasma proteins.

This measure is fundamental, because blood circulation is the major vehicle of drug distribution into human body

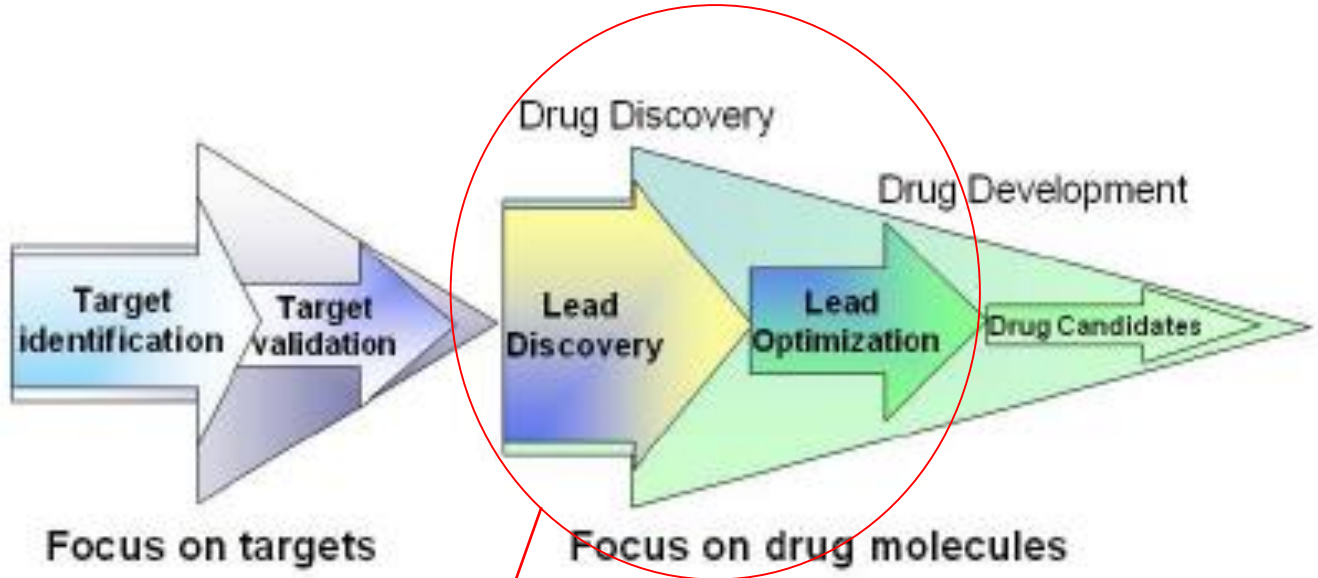


Drug Discovery Process



- What does the drug have to do?
- Which organs does it have to influence?
- What does the drug have NOT to do?
- Which organs does it have NOT to influence?

Drug Discovery Process

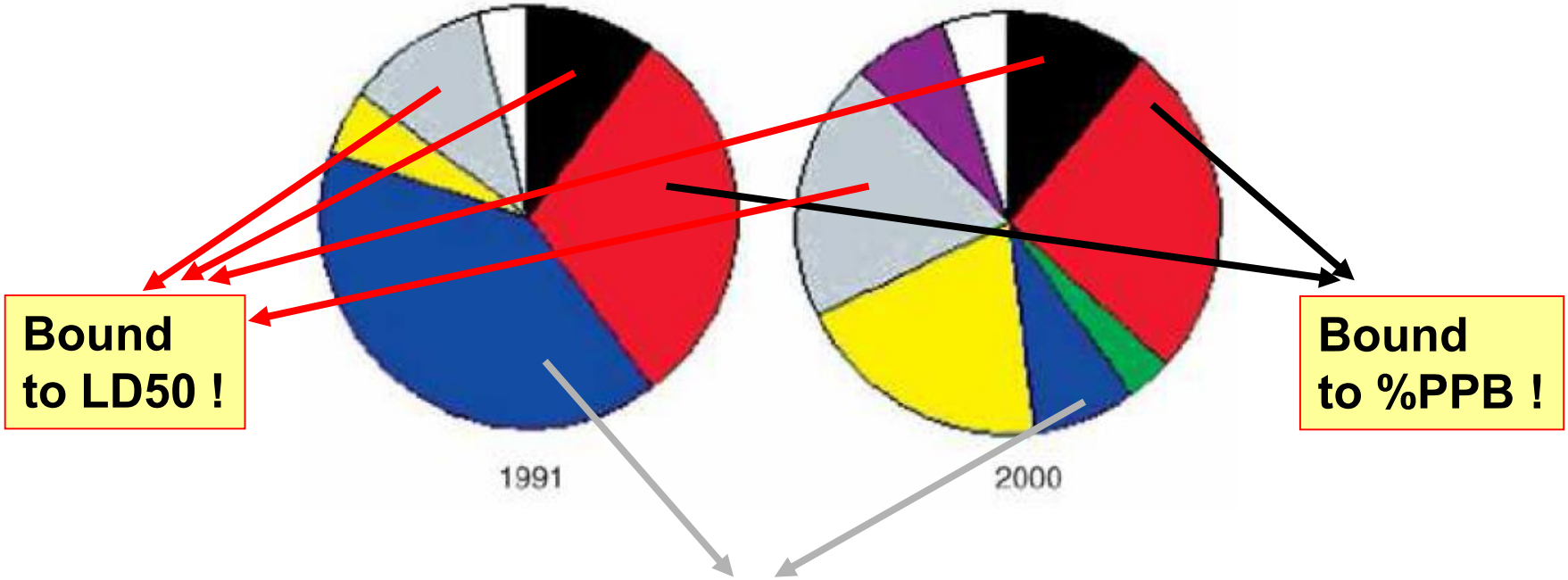


- Decide the desired value of some pharmacokinetic parameters
- Find the set of molecules that optimize them!

Predicting the values of pharmacokinetic parameters is ***crucial*** for the success/failure of the drug discovery process!!

Why are these parameters important?

Main reasons for failure in drug development:



- clinical safety (black)
- efficacy (red)
- formulation (green)
- bioavailability (blue)

- commercial (yellow)
- toxicology (grey)
- cost of goods (purple)
- others (white)

Q.S.A.R. Approach

$$H = \left(\begin{array}{cccc|c} x_{11} & x_{12} & x_{13} & \dots & x_{1M} & y_1 \\ x_{21} & x_{22} & x_{23} & \dots & x_{2M} & y_2 \\ \dots & & & & & \dots \\ x_{N1} & x_{N2} & x_{N3} & \dots, & x_{NM} & y_N \end{array} \right) \} \text{ drugs}$$

molecular descriptors

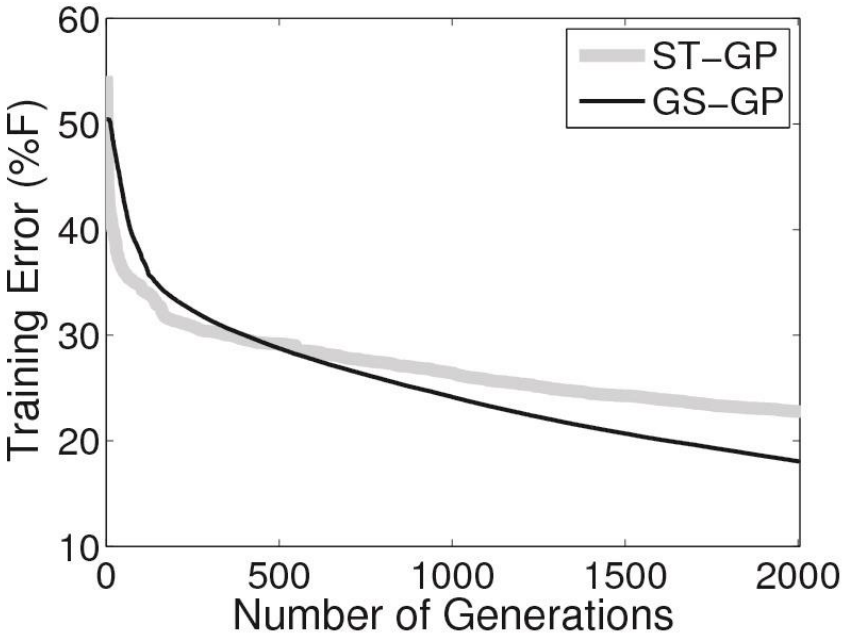
goal (or target - the value of the pharmacokinetic parameter to estimate)

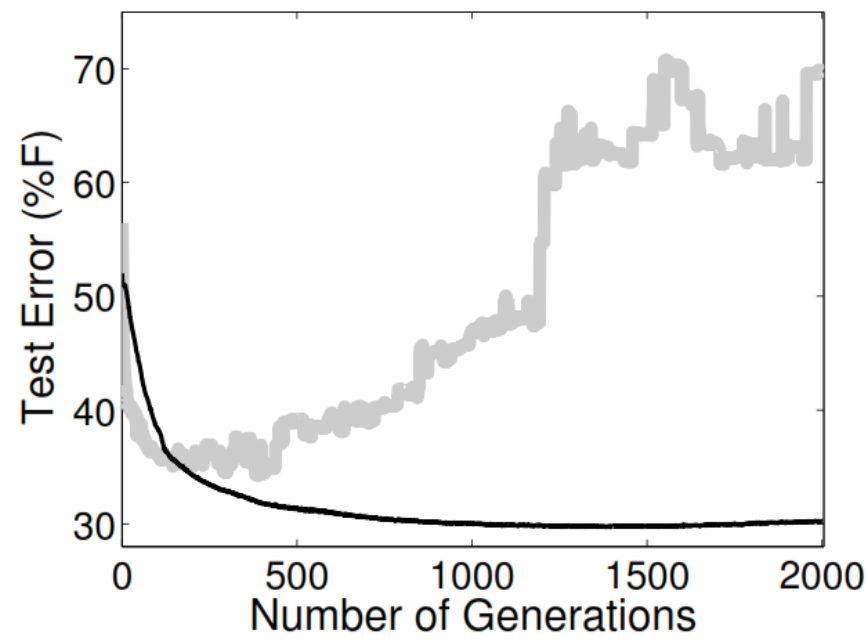
We look for a function *f* such that:

$$\forall\, i: 1 \le i \le N \quad f(x_{i1}, x_{i2}, x_{i3}, \dots, x_{iM}) = y_i$$

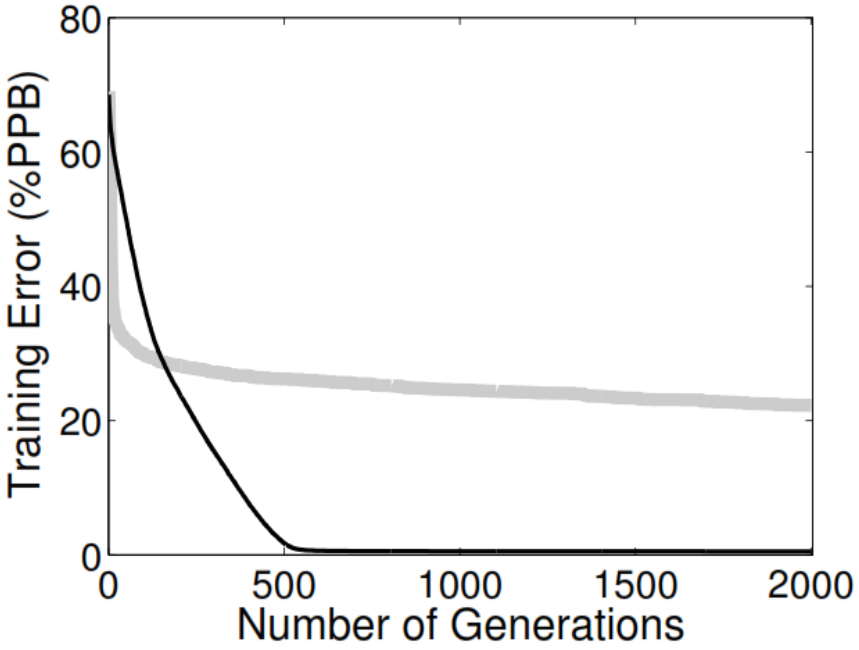
*... and it has to have a high **generalization** ability !*

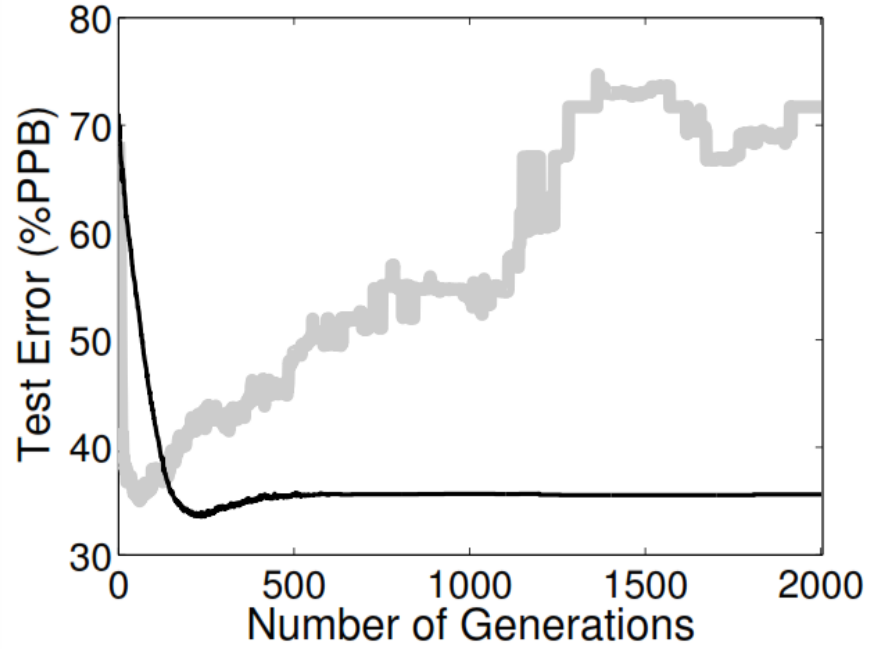
Bioavailability Results



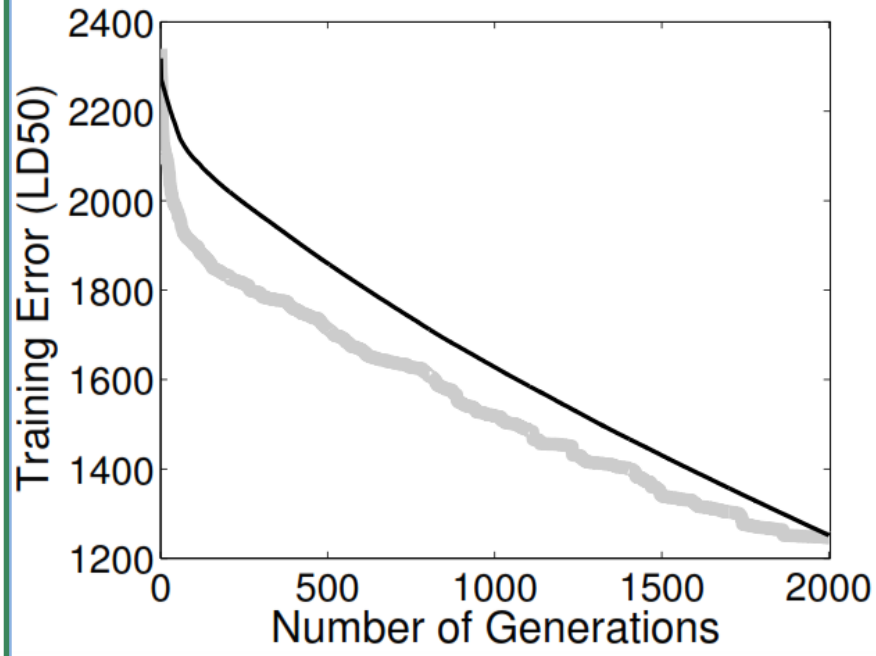


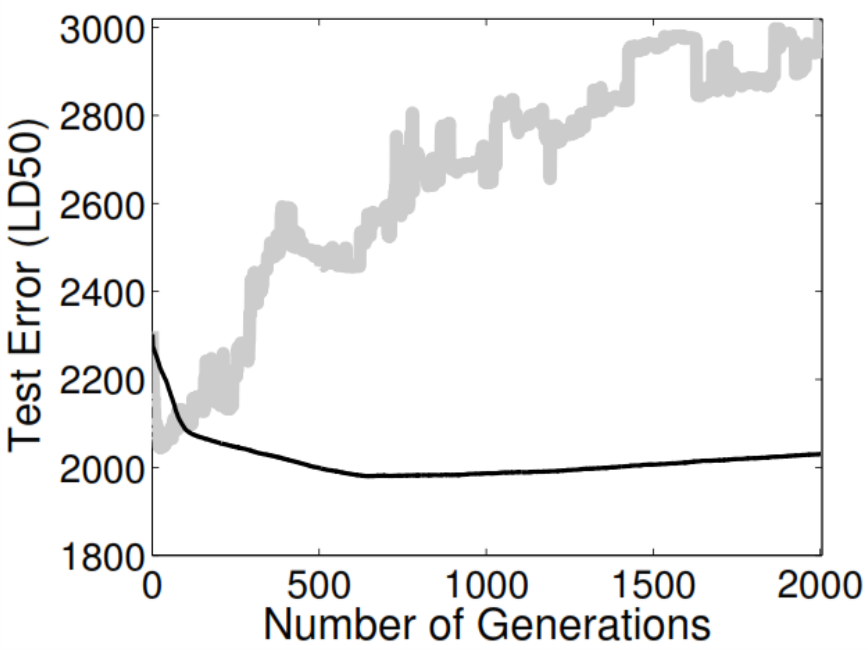
Plasma Protein Binding Level results





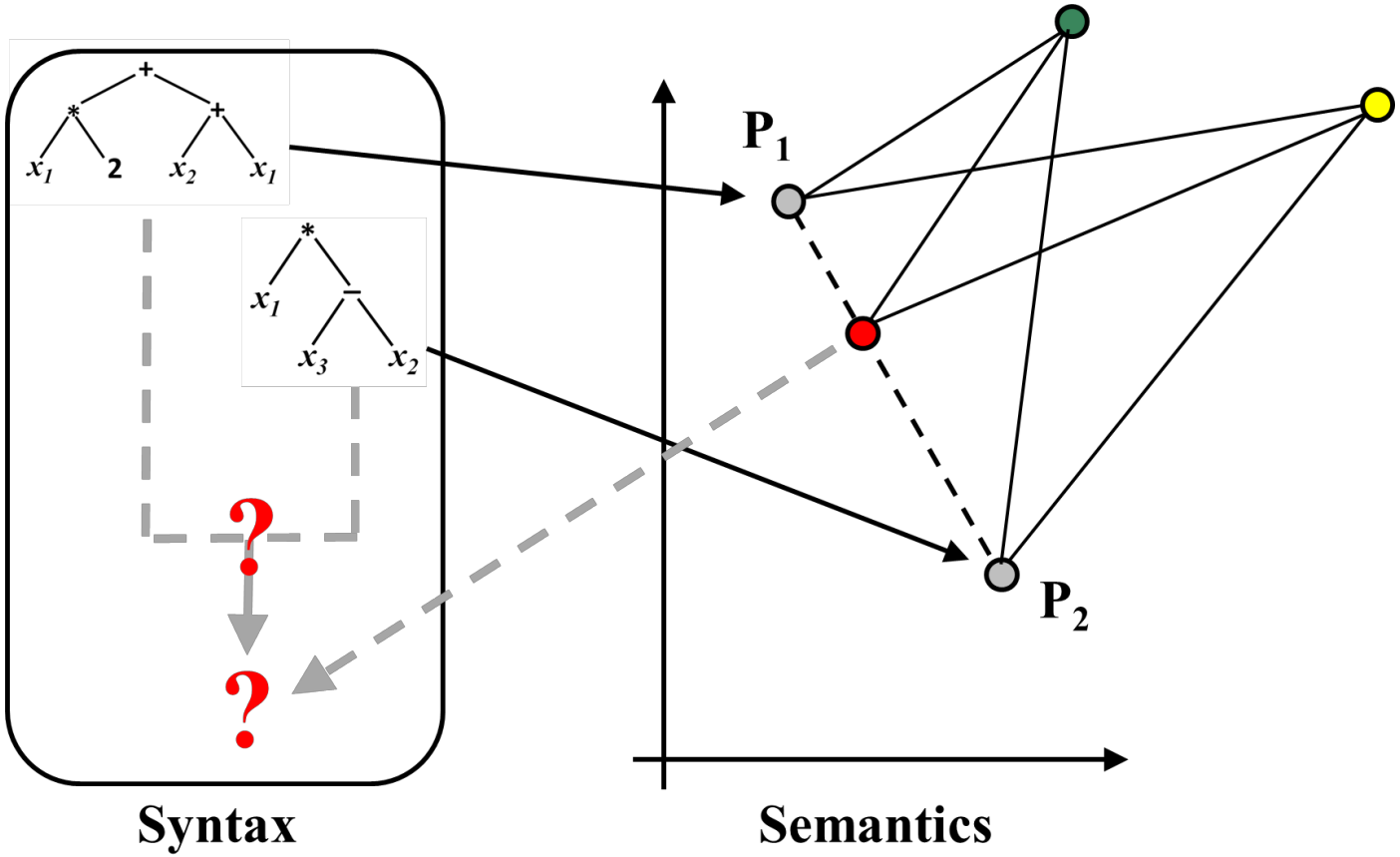
Toxicity Results





Why this good results on test data?

The geometrical properties of Moraglio's operators hold independetly of the dataset on which individuals are evaluated !



Controlling Overfitting

Even though geometric semantic operators do **not** guarantee an **improvement** on test data at each application, at least...

... they guarantee that the eventual worsening on test data is **limited**, and limited of a well defined quantity:

- For crossover:
  the fitness (on test data) of the worst parent
- For mutation:
  the mutation step *ms*

Prediction of the Unified Parkinson's Disease Rating Scale Assessment

M. Castelli, L. Vanneschi, and S. Silva. Prediction of the unified Parkinson's disease rating scale assessment using a genetic programming system with geometric semantic genetic operators. *Expert Systems with Applications*, 41(10):4608 – 4616, 2014

Unified Parkinson's Disease Rating Scale

The Unified Parkinson's Disease Rating Scale (UPDRS) is a scale that was developed as an effort to incorporate elements from existing scales to provide a comprehensive, efficient and flexible way of measuring and monitoring Parkinson's Disease (PD)-related disability and impairment.

For many persons affected by PD, the necessary specialized medical examinations to estimate the severity of their symptoms are difficult and invasive and they have to be performed by trained medical staff. This highlights the need of reliable and accurate computational techniques that allow estimating the UPDRS automatically and effectively.

Datasets

- Total-UPDRS: dataset using as target the values of the severity of the general PD symptoms, taking into account:
  - Mentation, Behavior and Mood.
  - Activities of daily living.
  - Motor.

  It reflects the presence and severity of symptoms, expressing it in a range from 0 to 176, with 0 representing a healthy state and 176 total disability.

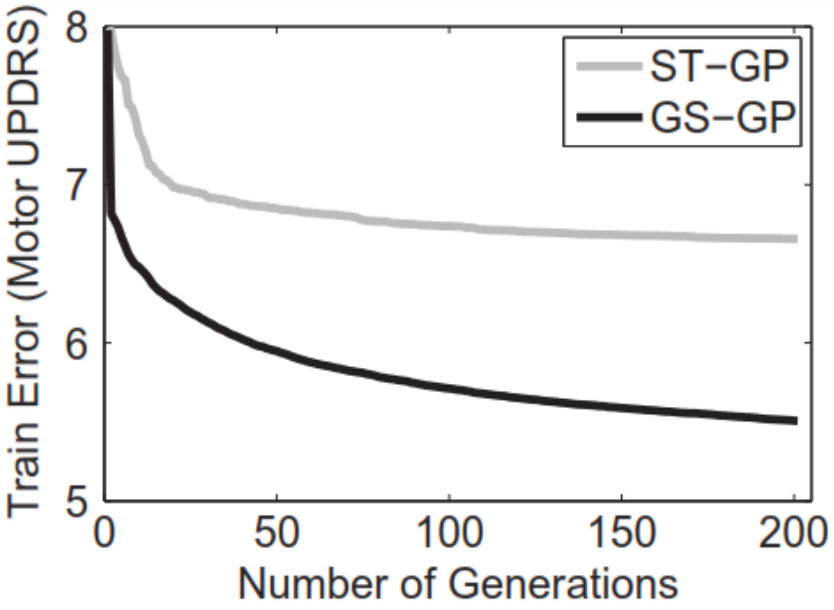
- Motor-UPDRS: dataset using as target the values of the severity of the motor symptoms.
  The motor section of the UPDRS encompasses tasks such as speech, facial expression, tremor and rigidity and expresses the severity of the related symptoms in a range from 0 to 108, where 0 represents a symptom free state and 108 denotes severe motor impairment.

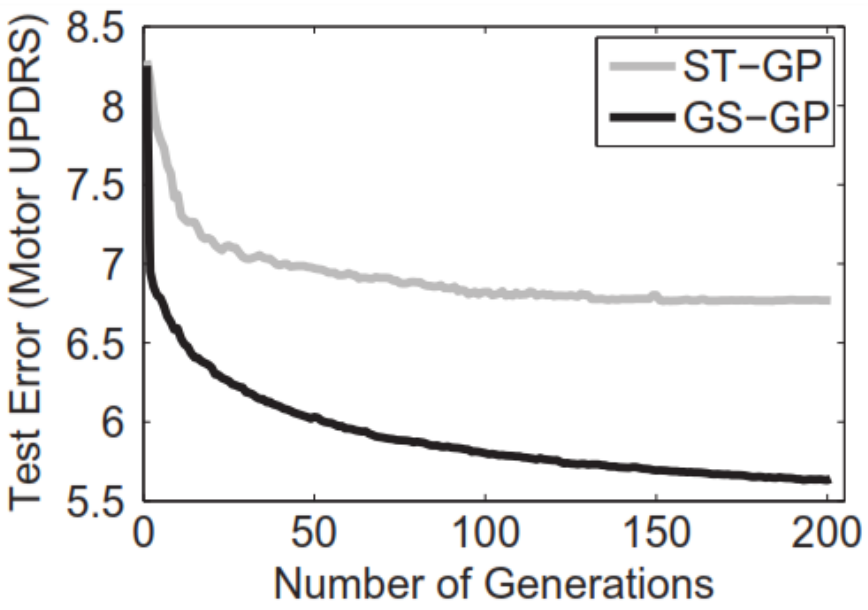
Feature

Feature	Description
Age	Subject age
Sex	Subject gender "0" – male, "1" – female
MDVP:Jitter (ABS)	Kay Pentax MDVP absolute jitter in microseconds (Boersma, 2001)
MDVP:RAP	Kay Pentax MDVP relative amplitude perturbation (Boersma, 2001)
MDVP:PPQ	Kay Pentax MDVP five-point period perturbation quotient (Boersma, 2001)
Jitter:DDP	Average absolute difference of differences between cycles, divided by the average period (Boersma, 2001)
MDVP:Shimmer	Kay Pentax MDVP local shimmer (Boersma, 2001)
MDVP:Shimmer (dB)	Kay Pentax MDVP local shimmer in decibels (Boersma, 2001)
Shimmer:APQ3	Three-point amplitude perturbation quotient (Boersma, 2001)
Shimmer:APQ5	Five-point amplitude perturbation quotient (Boersma, 2001)
MDVP:APQ	Kay Pentax MDVP 11 point amplitude perturbation quotient (Boersma, 2001)
Shimmer:DDA	Average absolute difference between consecutive differences between the amplitudes of consecutive periods (Boersma, 2001)
NHR	Noise to harmonies ratio (Boersma, 2001)
HNR	Harmonies to noise ratio (Boersma, 2001)
RPDE	Recurrence period density entropy (Little et al., 2007)
DFA	Detrended fluctuation analysis (Little et al., 2007)
D2	Correlation dimension (Kantz & Schreiber, 2004)
PPE	Pitch period entropy (Little et al., 2009)

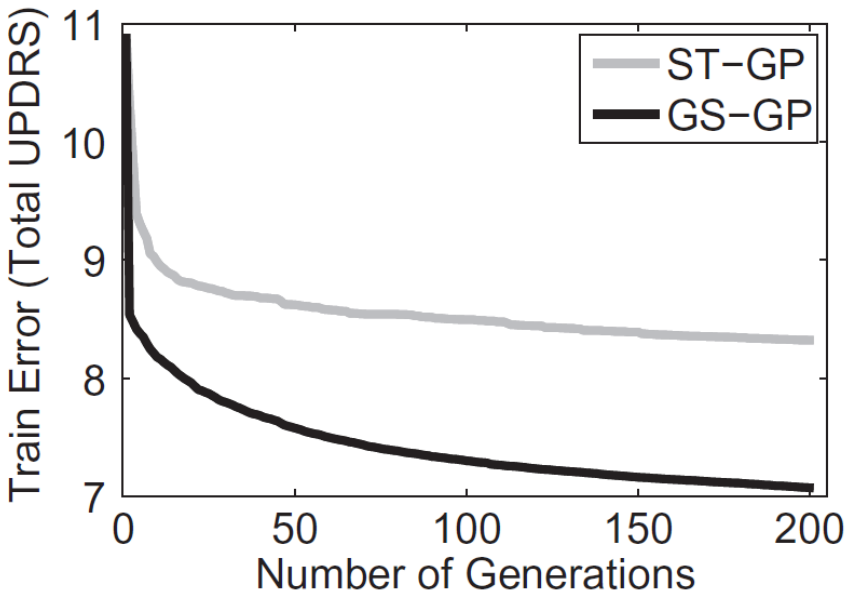
Features in the considered dataset. MDVP stands for (Kay Pentax) Multidimensional Voice Program

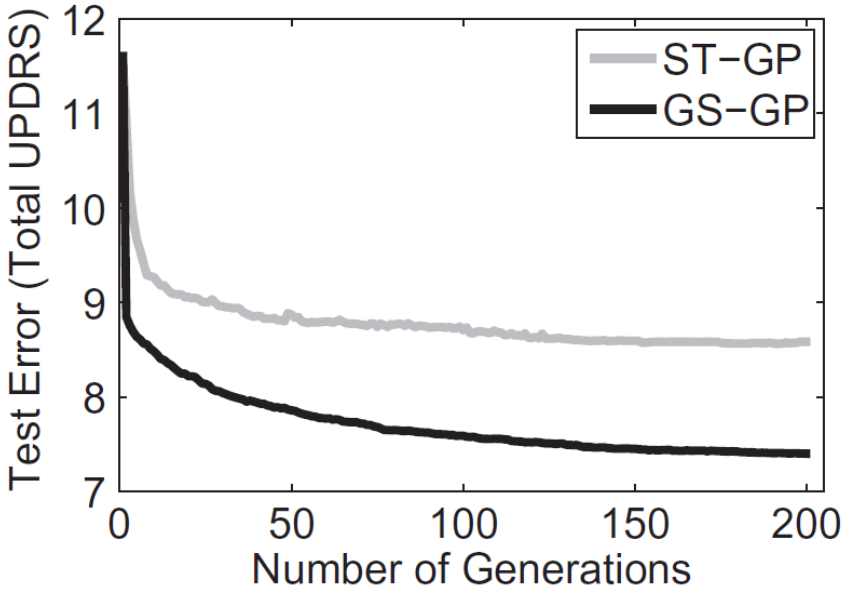
Parkinson's Desease Results
Motor Dataset



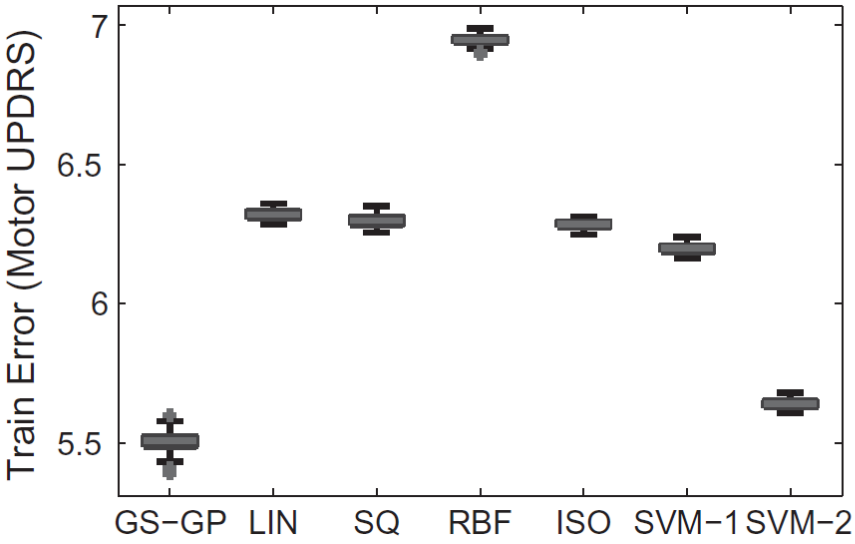


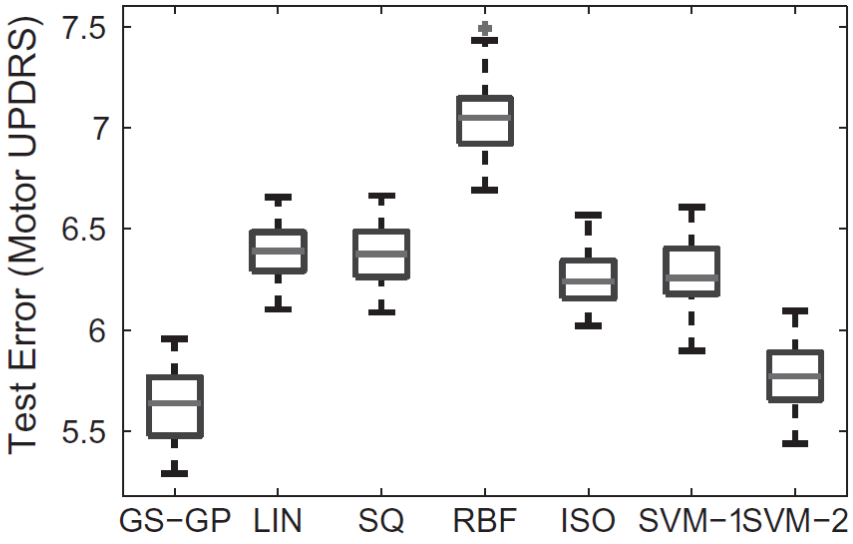
Parkinson's Desease Results
Total Dataset





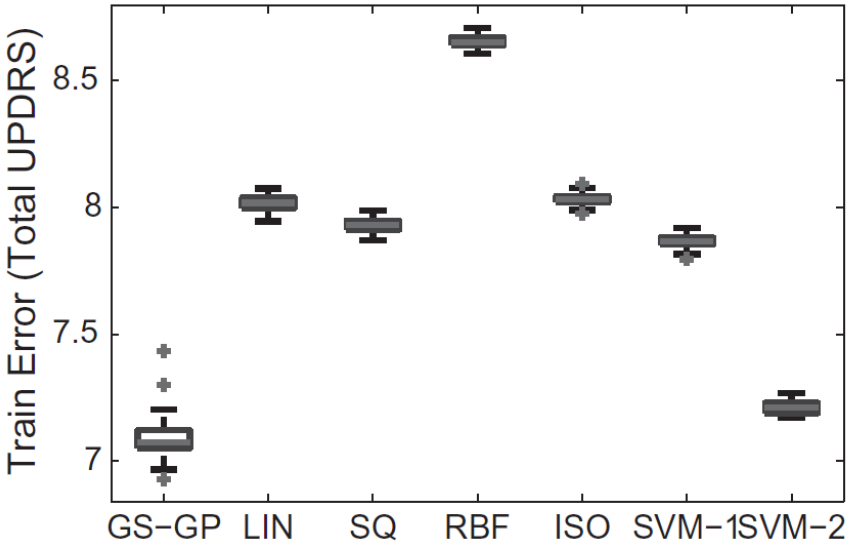
Comparison with Other ML Methods Motor Dataset

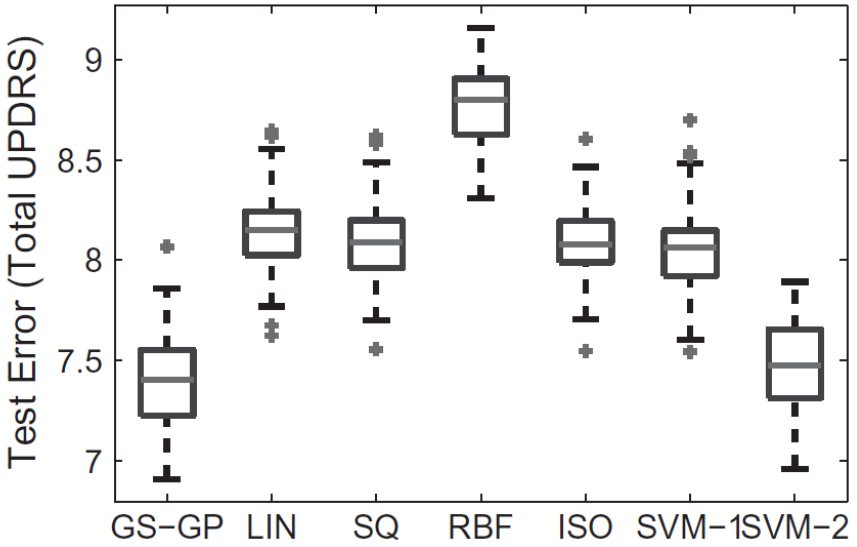




LIN -> regression (Weisberg, 2005)
SQ -> Least square regression (Seber & Wild, 2003)
RBF -> Radial Basis Function network (Haykin, 1999)
ISO -> Isotonic regression (Hoffmann, 2009)
SVM-1 -> SVM polynomial kernel of degree 1 (Ikopf & Smola, 2002)
SVM-2 -> SVM polynomial kernel of degree 2 (Ikopf & Smola, 2002)

Comparison with Other ML Methods Motor Dataset





LIN -> regression (Weisberg, 2005)
SQ -> Least square regression (Seber & Wild, 2003)
RBF -> Radial Basis Function network (Haykin, 1999)
ISO -> Isotonic regression (Hoffmann, 2009)
SVM-1 -> SVM polynomial kernel of degree 1 (Ikopf & Smola, 2002)
SVM-2 -> SVM polynomial kernel of degree 2 (Ikopf & Smola, 2002)

Statistical Comparison

		Motor-UPDRS					
		LIN	SQ	RBF	ISO	SVM-1	SVM-2
GS-GP	TRAIN	7.06E−18	7.06E−18	7.07E−18	7.07E−18	7.06E−18	7.07E−18
	TEST	7.07E−18	7.07E−18	7.07E−18	7.07E−18	8.99E−18	7.70E−05

		Total-UPDRS					
		LIN	SQ	RBF	ISO	SVM-1	SVM-2
GS-GP	TRAIN	7.06E−18	7.06E−18	7.07E−18	7.06E−18	7.06E−18	5.02E−14
	TEST	3.74E−17	7.12E−17	7.07E−18	8.00E−17	2.40E−16	3.58E−02

As a first step, the Kolmogorov–Smirnov test has shown that the data are not normally distributed. Then, the Wilcoxon rank-sum test for pairwise data comparison has been used under the alternative hypothesis that the samples do not have equal medians.
Used significance level was $\alpha = 0.01$.
Bonferroni correction was used.

Prediction of high performance concrete strength

M. Castelli, L. Vanneschi, and S. Silva. Prediction of high performance concrete strength using genetic programming with geometric semantic genetic operators. *Expert Systems with Applications*, 40(17):6856 – 6862, 2013.

The Problem

Concrete is the most-used man-made product in the world.

Concrete is a composite construction material made primarily with aggregate, cement, and water.

Several way of mixing these ingredients, and several ways of establishing that the concrete will have the required performance in different situations.

Reliable and accurate techniques that allow modelling the behaviour of concrete materials are in demand.

Data

The dataset consist in 1028 instances, each of them described by 8 variables:

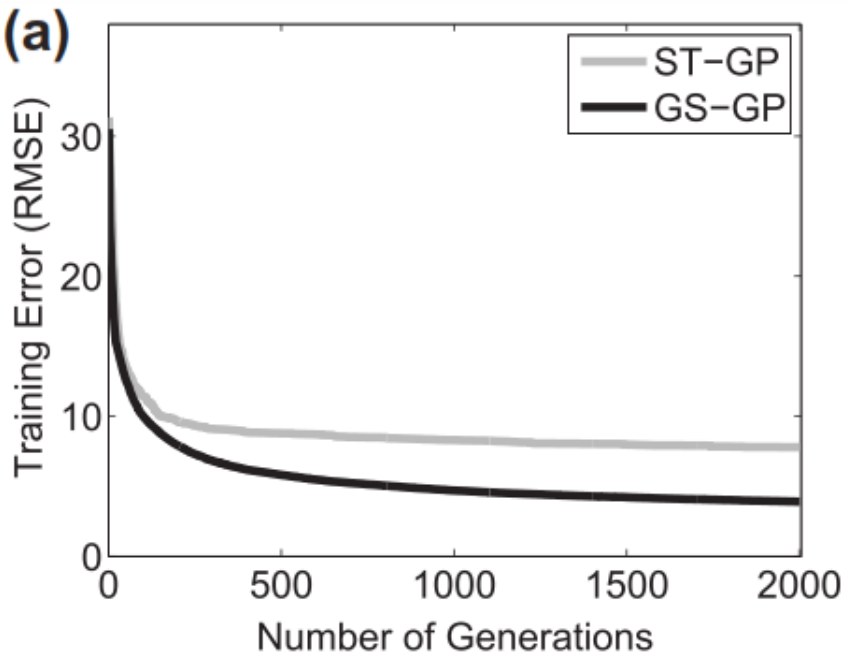
Variables used to describe each instance. For each variable minimum, maximum, average, median and standard deviation values have been reported.

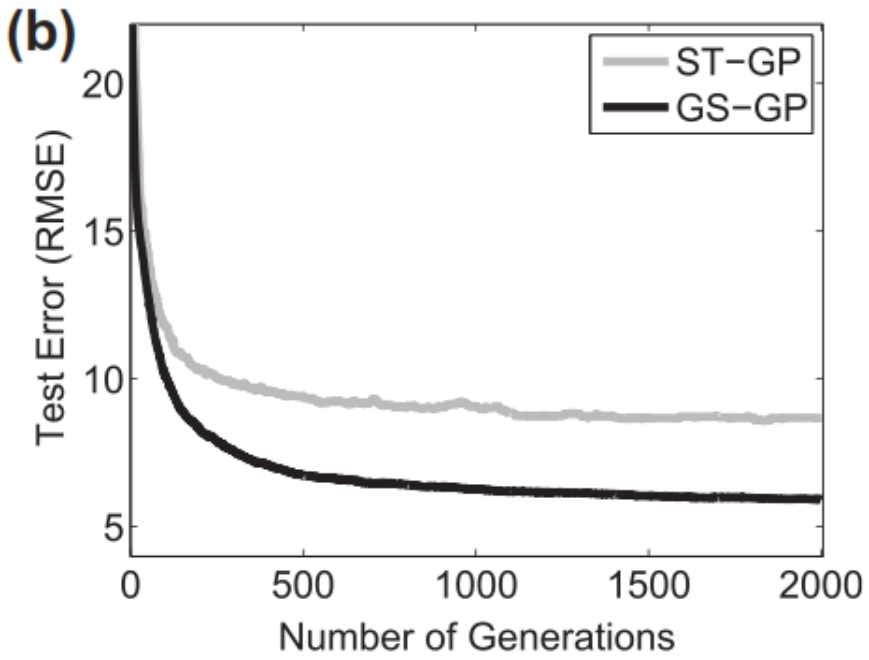
	Minimum	Maximum	Average	Median	Standard Deviation
Cement (kg/m^3)	102.0	540.0	281.2	272.9	104.5
Fly ash (kg/m^3)	0.0	359.4	73.9	22.0	86.3
Blast furnace slag (kg/m^3)	0.0	200.1	54.2	0.0	64.0
Water (kg/m^3)	121.8	247.0	181.6	185.0	21.4
Superplasticizer (kg/m^3)	0.0	32.2	6.2	6.4	6.0
Coarse aggregate (kg/m^3)	801.0	1145.0	972.9	968.0	77.8
Fine aggregate (kg/m^3)	594.0	992.6	773.6	779.5	80.2
Age of testing (days)	1.0	365.0	45.7	28.0	63.2

Dataset from:

Yeh, I.-C. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research*, 28(12), 1797–1808. 1998.

Results


(a)
ST−GP
GS−GP
Training Error (RMSE)
30
20
10
0
0
500
1000
1500
2000
Number of Generations


(b)
ST−GP
GS−GP
Test Error (RMSE)
20
15
10
5
0
500
1000
1500
2000
Number of Generations

Comparison with other ML Methods

Concrete Strength prediction:

Method	Train	Test
Linear regression (Weisberg, 2005)	10.567	10.007
Square Regression (Seber & Wild, 2003)	17.245	15.913
Isotonic Regression (Hoffmann, 2009)	13.497	13.387
Radial Basis Function Network (Haykin, 1999)	16.778	16.094
SVM Polynomial Kernel (1 degree) (lkopf & Smola, 2002)	10.853	10.260
SVM Polynomial Kernel (2 degree)	7.830	7.614
SVM Polynomial Kernel (3 degree)	6.323	6.796
SVM Polynomial Kernel (4 degree)	5.567	6.664
SVM Polynomial Kernel (5 degree)	4.938	6.792
Artificial Neural Networks (Haykin, 1999)	7.396	7.512
Standard GP	7.792	8.67
Geometric Semantic GP	3.897	5.926

Prediction of the Relative Position of Computer Tomography (CT) Slices

M. Castelli, L. Trujillo, L. Vanneschi, and A. Popovic. Prediction of relative position of CT slices using a computational intelligence system. *Applied Soft Computing*. Volume 6. Pages 537-542. 2016. doi:http://dx.doi.org/10.1016/j.asoc.2015.09.021

The Problem

Scanning large parts of a patient's body with computerized tomography (CT) is common practice in radiology.

The amount of image data resulting from a full body scan varies between 40MB to more than 1GB, which has to be stored in a medical picture archiving and communication system (PACS).

A clinician often needs to compare different scans of the same body region for differential diagnoses or needs to compare disease patterns in the same body region between similar patients.

When searching for similar disease patterns in a remote PACS it is impractical to load complete volume sets and then manually navigate to the relevant body regions. Instead, transferring only the most similar body parts is more reasonable, secure and ecient. Nonetheless, in both scenarios it is necessary to determine the relative position of the given CT slice within the body.

Integration of GSGP with Local Search

GSGP's optimization is very effective, but quite slow.

We recently proposed a GSM operator revisited:

$$T_M = \alpha_0 + \alpha_1 \cdot T + \alpha_2 \cdot (T_{R1} + T_{R2})$$

where $\alpha_i \in \Re$ andα_2 corresponds to the original mutation step *ms.* Given a training set of n fitness cases, this defines system with n linear equations and three unknowns $(\alpha_0, \alpha_1, \alpha_2)$.

This is an overdetermined multivariate linear fitting problem, which can be solved using Singular Value Decomposition (SVD).

New Algorithm Variants

- GSGP
- GSGP-LS (using GSM-LS)
- HYBRID
  - Run GSGP-LS for the first M generations/ run standard GSGP afterwards (Best results found with M=10)

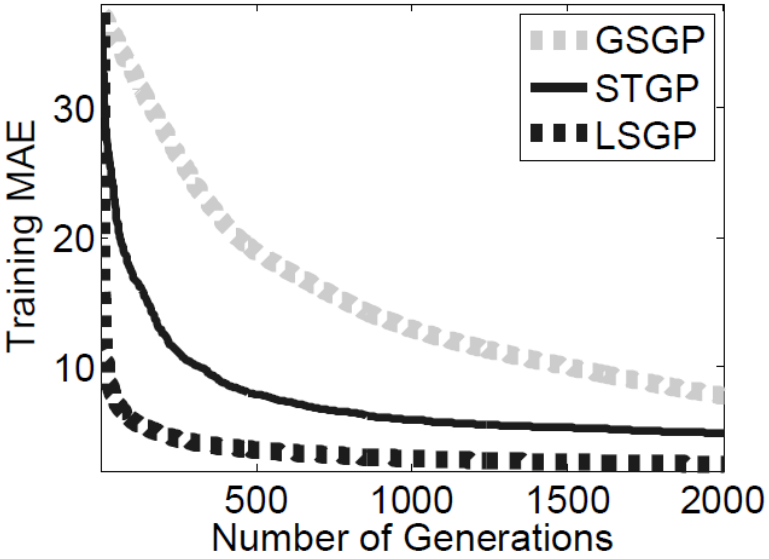
Data

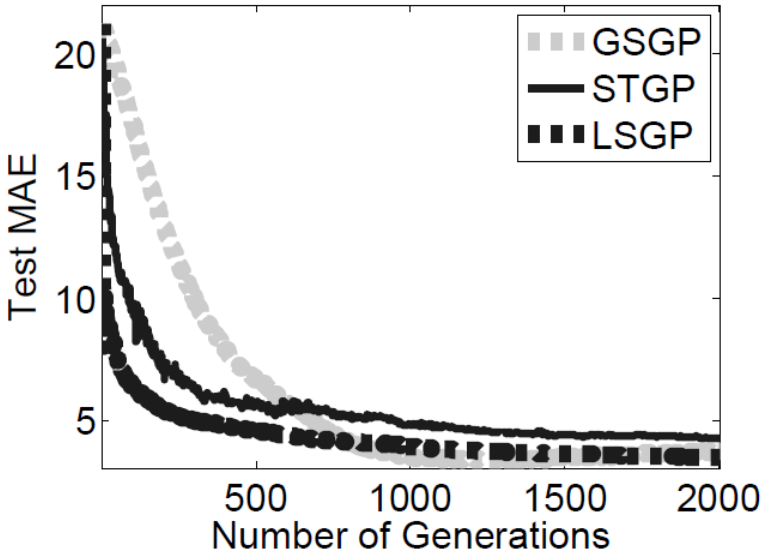
The data was retrieved from 53500 CT images taken from 74 different patients (43 male, 31 female).

Each CT image is described by 385 features. The first feature is the ID of the patient; features 2 – 241 are related to bone structures; features 242 - 385 are related to air inclusions. The last feature is the target variable that is the relative location of the image on the axial axis.

Values are in the range [0; 180] where 0 denotes the top of the head and 180 the soles of the feet.

Data





	Testing		Training	
	STGP	LSGP	STGP	LSGP
GSGP	0.065	1.48E-6	2.60E-11	2.60E-11
LSGP	0.003	-	2.60E-11	-

Comparison with Other Methods

Method	Training	Test	Improvement
Neural Networks [23]	17.08	15.32	77%
Isotonic Regression [24]	9.91	12.41	72%
Best method proposed in [1]	N.A.	4.45	22%
STGP	4.89	4.25	19%
GSGP	7.8	3.70	7.5%
LSGP	**2.52**	**3.44**	-

Forecasting of Energy Consumption

M. Castelli, L. Vanneschi, and M. De Felice. Forecasting short-term electricity consumption using a semantics-based genetic programming framework: The south Italy case. *Energy Economics*, 47:37 – 41, 2015

M. Castelli, L. Trujillo, and L. Vanneschi. Energy consumption forecasting using semantic based genetic programming with local search optimizer. *Computational Intelligence and Neuroscience*, Volume 57, 2015. Article ID 971908. http://dx.doi.org/10.1155/2015/971908.

M. Castelli, L. Trujillo, L. Vanneschi, and A. Popovic. Prediction of energy performance of residential buildings: A genetic programming approach. *Energy and Buildings*, 102:67 – 74, 2015.

The Problem

The aim of this forecasting task is to predict the energy consumption at day t, providing information until day $t-1$ (one-day ahead forecasting) using the past samples of the load and weather information.

Data

Historical energy consumption data and weather information in Italy in the years between 1999 and 2010 have been used to test the performance of the proposed system.

Data include temperatures, pressure values, wind speed, and other weather related information.

Data from 1999 to 2006 have been used during the training phase, while the remaining available data (i.e., from 2006 to 2010) have been used to validate the model on unseen data and hence to assess the quality of the forecasting.

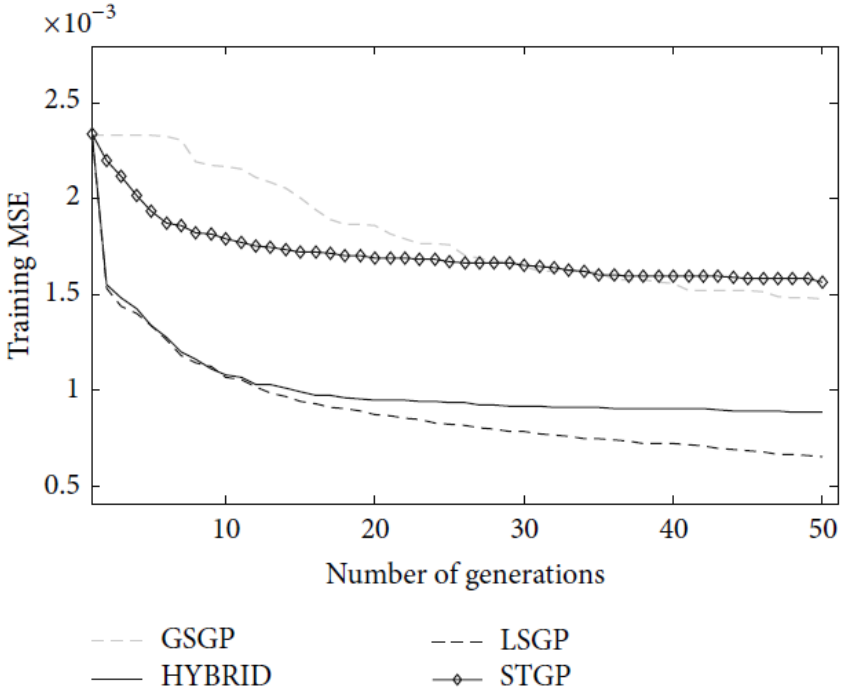
Data Provider

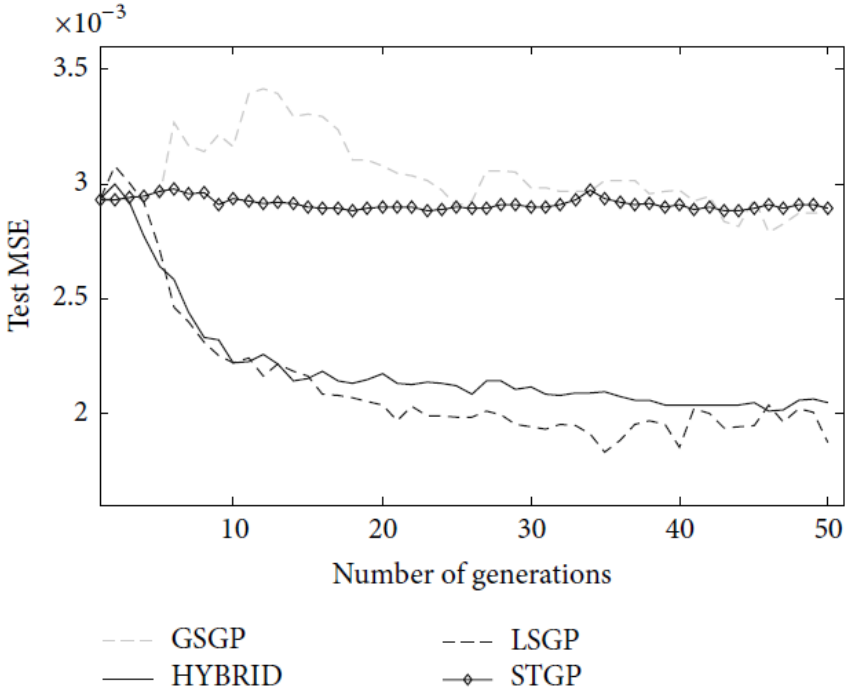
TERNA S.p.A. (Rete Elettrica Nazionale) is an Italian electricity transmission system operator based in Rome, Italy.

With 63,500 kilometres of power lines or around 98% of the Italian high-voltage power transmission grid, TERNA is the first independent electricity transmission grid operator in Europe and the sixth in world based on the size of its electrical grid.

TERNA is the owner of the Italian transmission grid and responsible for energy transmission and dispatching.

Results


×10^−3
2.5
2
1.5
1
0.5
Training MSE
10
20
30
40
50
Number of generations
GSGP
LSGP
HYBRID
STGP


×10^−3
3.5
3
2.5
2
Test MSE
10
20
30
40
50
Number of generations
GSGP
LSGP
HYBRID
STGP

Comparison with Other ML Methods

Method	Training error	Test error
Linear regression Weisberg (2005)	0.67	0.75
Least square regression Seber and Wild (2003)	0.61	0.78
Radial basis function network Haykin (1999)	0.47	0.66
Isotonic regression Hoffmann (2009)	0.64	0.76
SVM polynomial kernel (degree 1) Schölkopf and Smola (2002)	0.53	0.68
SVM polynomial kernel (degree 2)	0.49	0.71
GS-GP	0.39	0.59

PART III

Conclusions

Summary of the contributions

- An *efficient implementation* of geometric semantic operators, that has allowed us to use them on real-life applications.
- *Excellent results* on the studied *applications*.
- *New insights about the generalization* ability of geometric semantic operators (without the novel implementation that allowed us to use geometric semantic GP on these complex real-life problems, this interesting property would probably remain unnoticed).

Open issues

The *reconstruction of the expression of the best individual*, even though we do it only once and after the termination of the run, is still an issue:

Individuals after hundreds of generations get so huge that it may be *impossible* to reconstruct their entire expression (even though it is possible to get some information about it, such as the features or primitives it uses...).

Models generated by geometric semantic GP are ***black*** (or at least ***"dark gray"*** ☺) boxes!

We are working on this!

All this would not exist without...

Mauro Castelli
NOVA IMS
Universidade Nova de Lisboa

Sara Silva
Faculdade de Ciências
Universidade de Lisboa

Thank you!

lvanneschi@novaims.unl.pt