Adding value to optimisation by interrogating fitness models

Alexander Brownlee

www.cs.stir.ac.uk/~sbr

sbr@cs.stir.ac.uk

Outline

- "Adding value"
- Markov network fitness model
- Single-generation examples (recap)
- Multi-generation examples
- Discussion
- (RW Application and some more discussion in SAEOpt tomorrow)

Value-added Optimisation

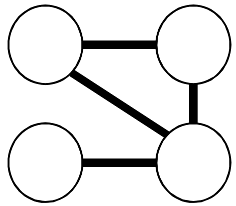
- A philosophy whereby we provide more than simply optimal solutions
- Information gained during optimisation can highlight sensitivities and linkage
- This can be useful to the decision maker:
  - Confidence in the optimality of results
  - Aids decision making
  - Insights into the problem
    - Help solve similar problems
    - Highlight problems / misconceptions in definition

Value-added Optimisation

- This information can come from
  - the trajectory followed by the algorithm
  - models built during the run
- If we are constructing a model as part of the optimisation process, anything we can learn from it comes "for free"
- See also
  - M. Hauschild, M. Pelikan, K. Sastry, and C. Lima. Analyzing probabilistic models in hierarchical BOA. IEEE TEC 13(6):1199-1217, December 2009
  - R. Santana, C. Bielza, J. A. Lozano, and P. Larranaga. Mining probabilistic models learned by EDAs in the optimization of multi-objective problems. In Proc. GECCO 2009, pp 445-452

Markov network fitness model (MFM)

- Originally developed as part of DEUM EDA
- An undirected probabilistic graphical model
  - Representation of the joint probability distribution (factorises as a Gibbs dist.)
  - Node: variables
  - Edges: dependencies between variables



- Gibbs distribution of MN is equated to mass distribution of fitness in population

$$p(x) = \frac{f(x)}{\sum_y f(y)} \equiv \frac{e^{-U(x)/T}}{\sum_y e^{-U(y)/T}}$$

$$\boxed{-\ln(f(x)) = U(x)/T}$$

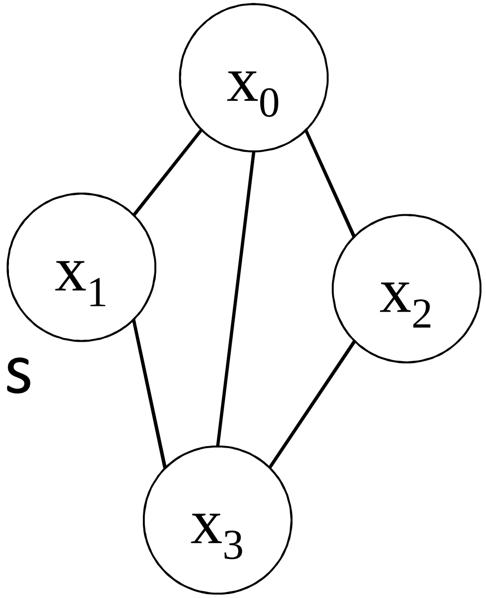
- Energy has negative log relationship to probability, so minimise U to maximise f

Markov network example

- For a bit-string encoded problem $f(x_0 \ldots x_3)$, model can be represented by:

$$\alpha_0 x_0 + \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 + \alpha_{01} x_0 x_1 + \alpha_{02} x_0 x_2 + \alpha_{03} x_0 x_3 \\ \alpha_{13} x_1 x_3 + \alpha_{23} x_2 x_3 + \alpha_{013} x_0 x_1 x_3 + \alpha_{023} x_0 x_2 x_3 + c = -\ln(f(x))$$

- Build a set of equations using values from population and solve to estimate the α
  - Variables are -1 and +1 instead of 0 and 1
- Can then sample to generate new solutions



Mining the model (1)

$$\boxed{-\ln(f(x)) = U(x)/T}$$

- As we minimise energy, we maximise fitness. So to minimise energy:

$$\alpha_i x_i$$

- If the value taken by x_i is 1 (+1) in high-fitness solutions, then a_i will be negative
- If the value taken by x_i is 0 (-1) in the high-fitness solutions, then a_i will be positive
- If no particular value is taken by x_i optimal solutions, then a_i will be near zero

Mining the model (2)

$$\boxed{-\ln(f(x)) = U(x)/T}$$

- As we minimise energy, we maximise fitness. So to minimise energy:

$$\alpha_{ij} x_i x_j$$

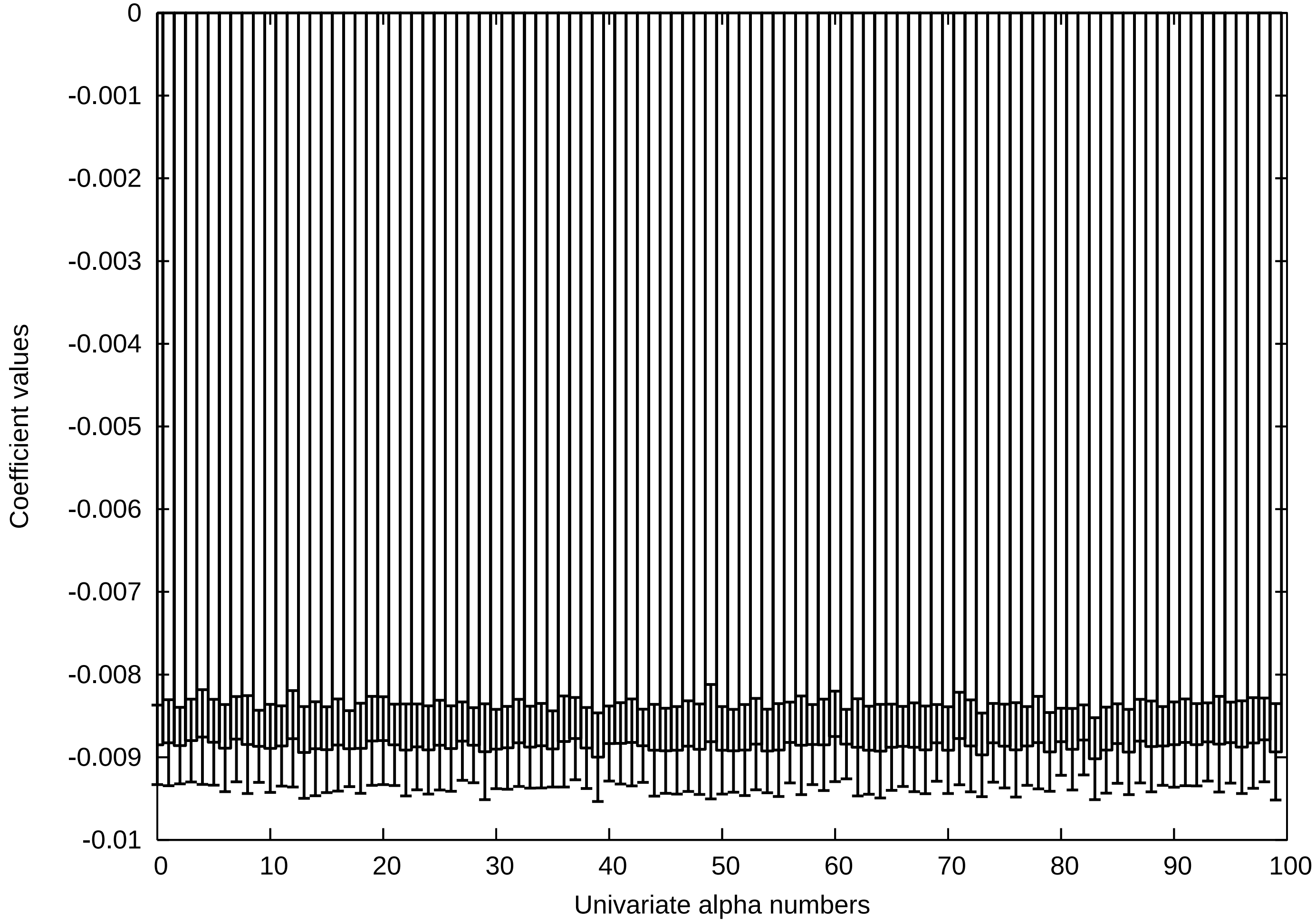
- If the values taken by x_i and x_j are equal (+1) in the optimal solutions, then a_i will be negative
- If the values taken by x_i and x_j are opposite (-1) in the optimal solutions, then a_{ij} will be positive
- Higher order interactions follow this pattern

Single stage experiments

- Often the model closely fits the fitness function in the first generation (see $DEUM_d$)
- Experiments:
  1. generate 30 populations of solutions at random and evaluate
  2. estimate MFM parameters for each population
  3. calculate means of each α across the 30 models
- This section mostly a recap of earlier results

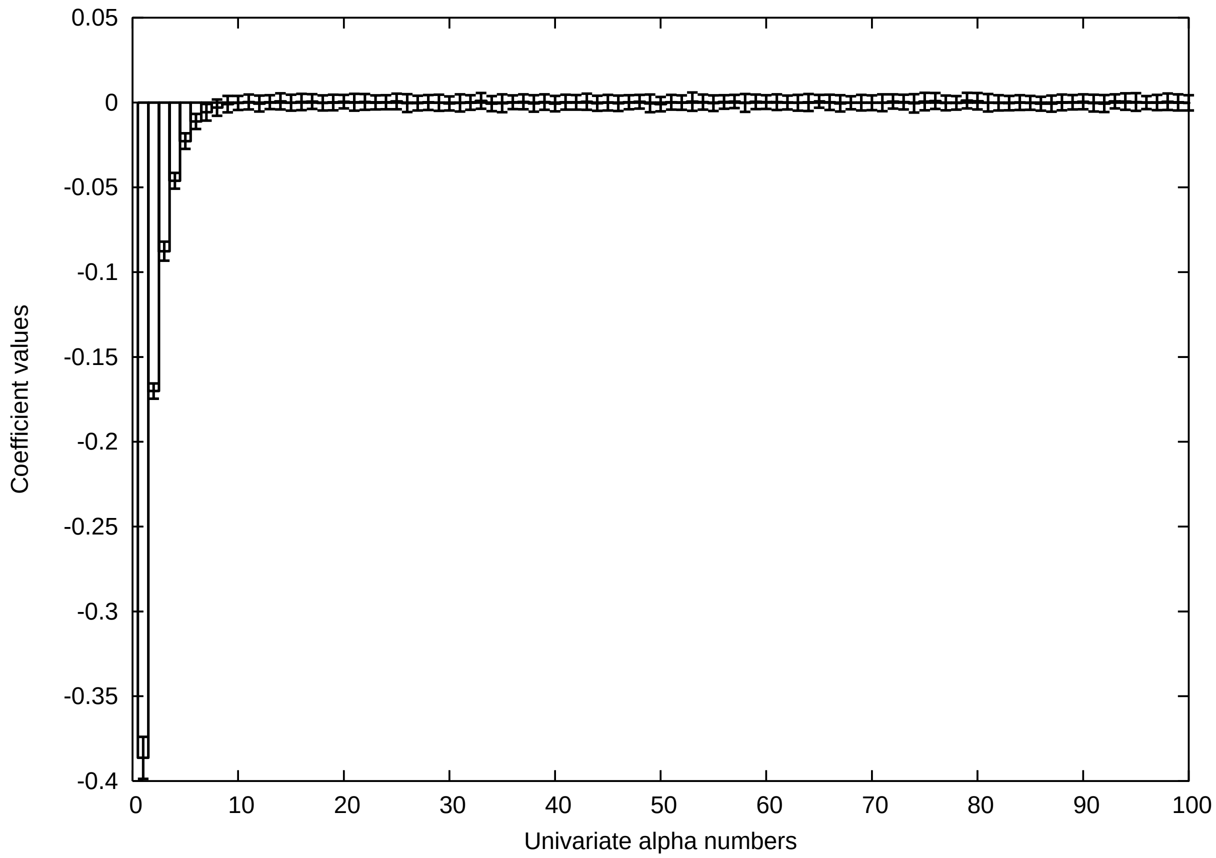
Onemax

- Fitness is the sum of x_i set to 1



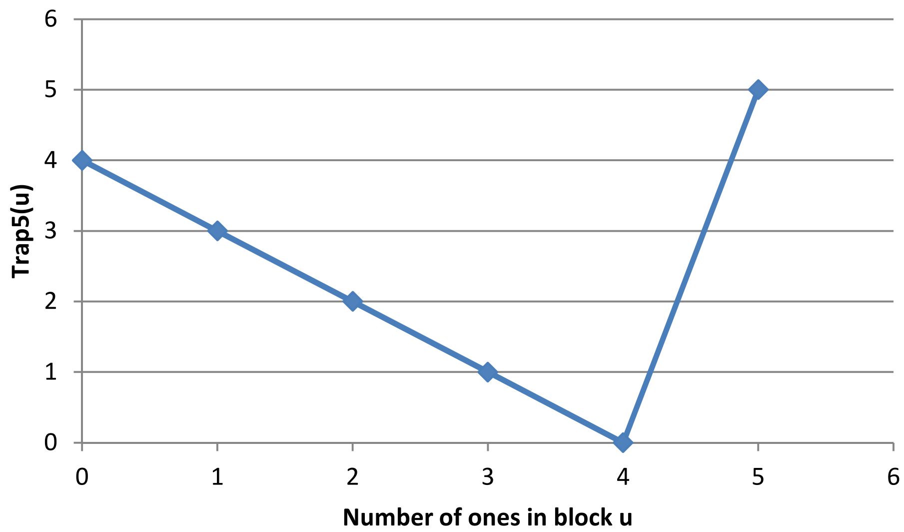
BinVal

- Fitness is the weighted sum of x_i set to 1 (the bit string is treated as a binary number)

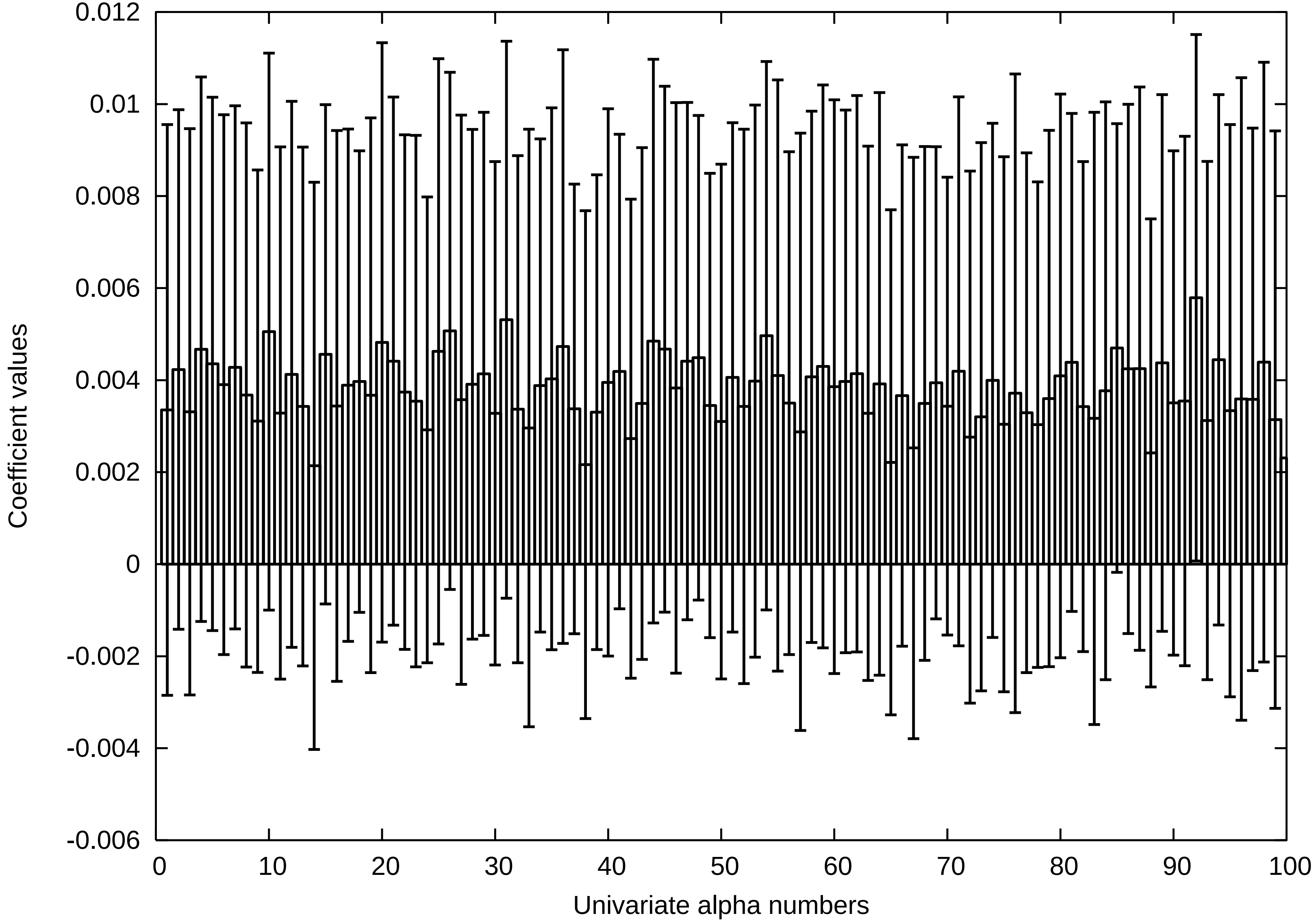


Trap 5

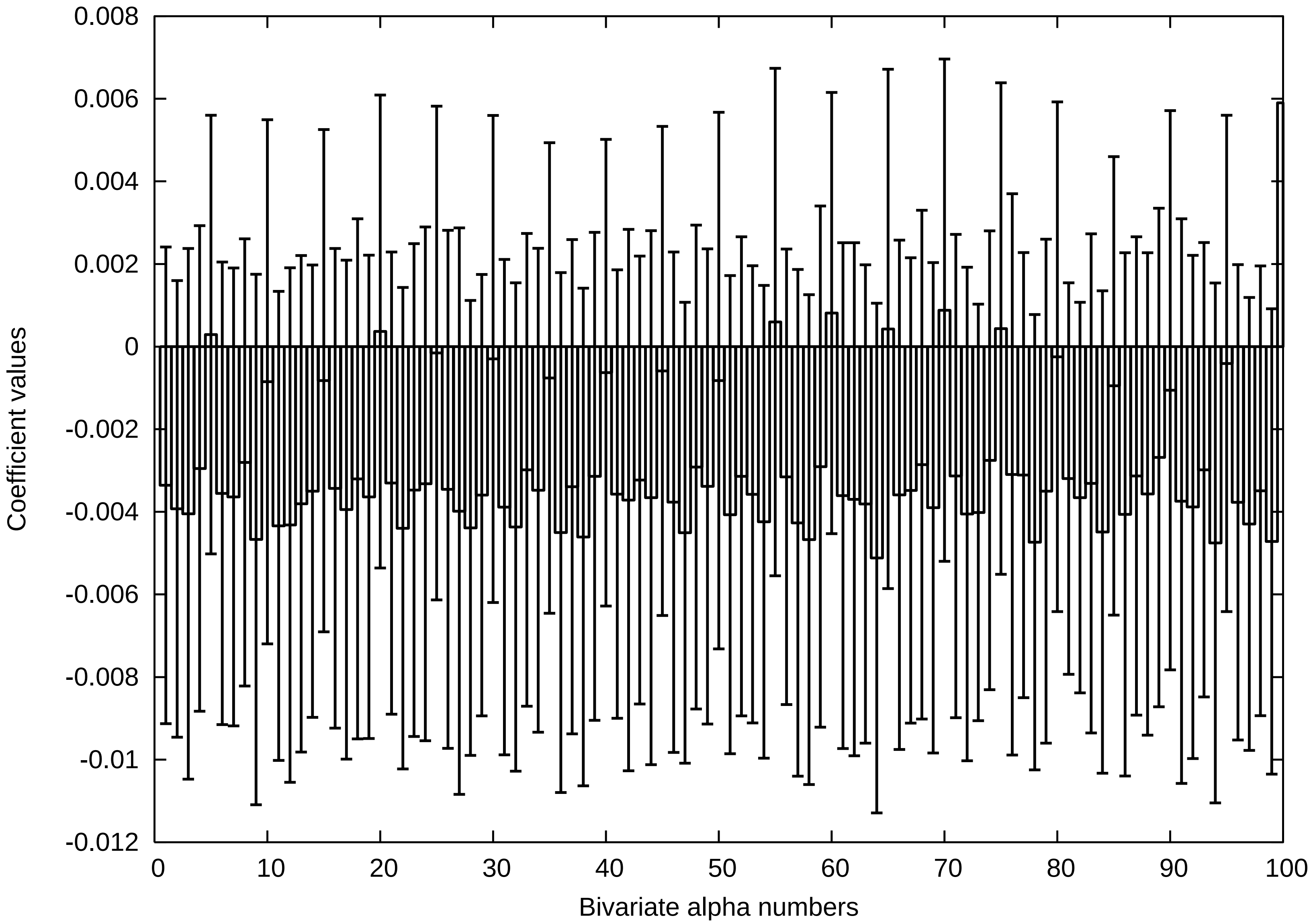
- Bit string is broken into blocks of size u
- The blocks are scored separately: fitness is sum of these scores
- Deceptive for algorithms ignoring the blocks



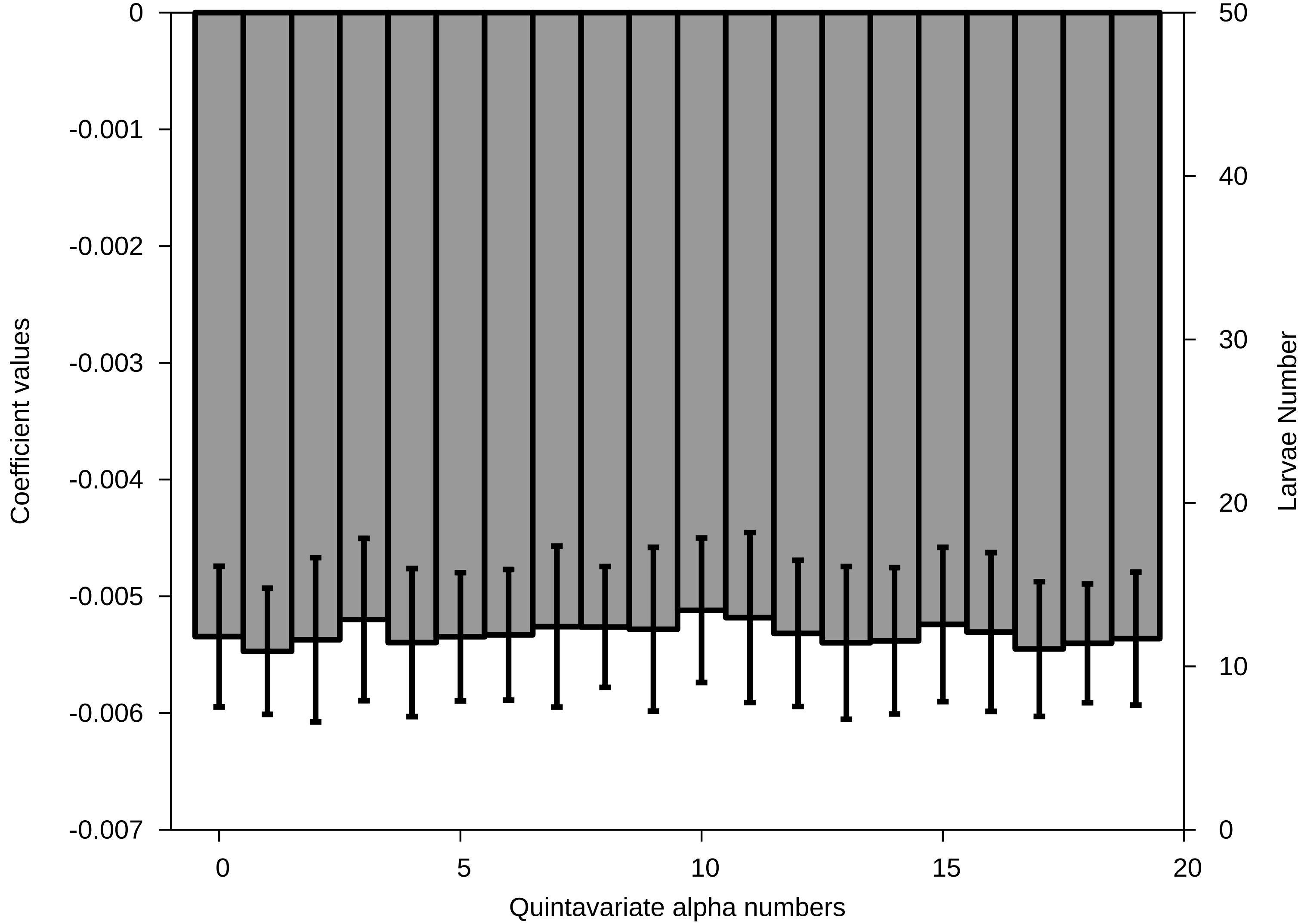
Trap 5


0.012
0.01
0.008
0.006
0.004
0.002
0
-0.002
-0.004
-0.006
Coefficient values
0
10
20
30
40
50
60
70
80
90
100
Univariate alpha numbers

Trap 5


0.008
0.006
0.004
0.002
0
-0.002
-0.004
-0.006
-0.008
-0.01
-0.012
Coefficient values
0
10
20
30
40
50
60
70
80
90
100
Bivariate alpha numbers

Trap 5



0
-0.001
-0.002
-0.003
-0.004
-0.005
-0.006
-0.007
Coefficient values
50
40
30
20
10
0
Larvae Number
0
5
10
15
20
Quintavariate alpha numbers

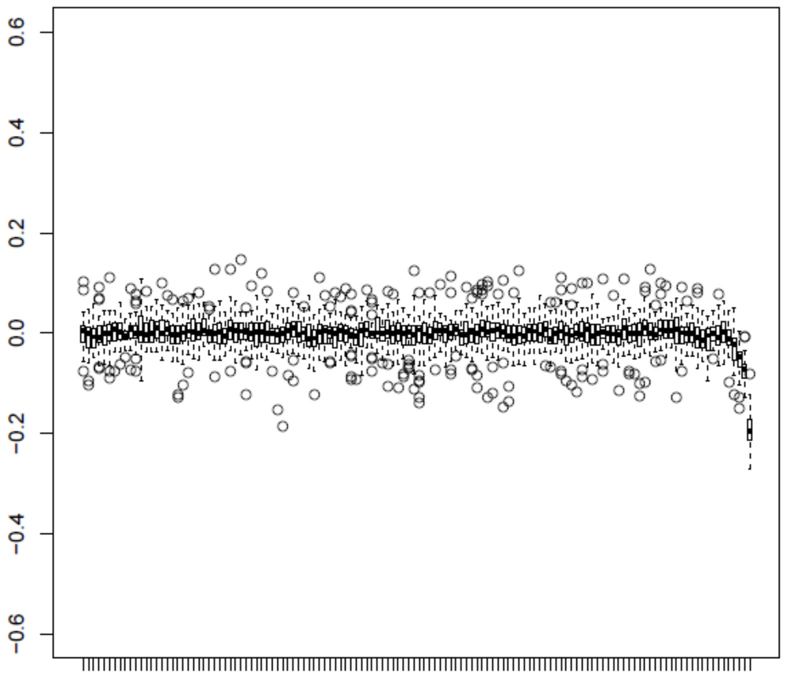
Experiments

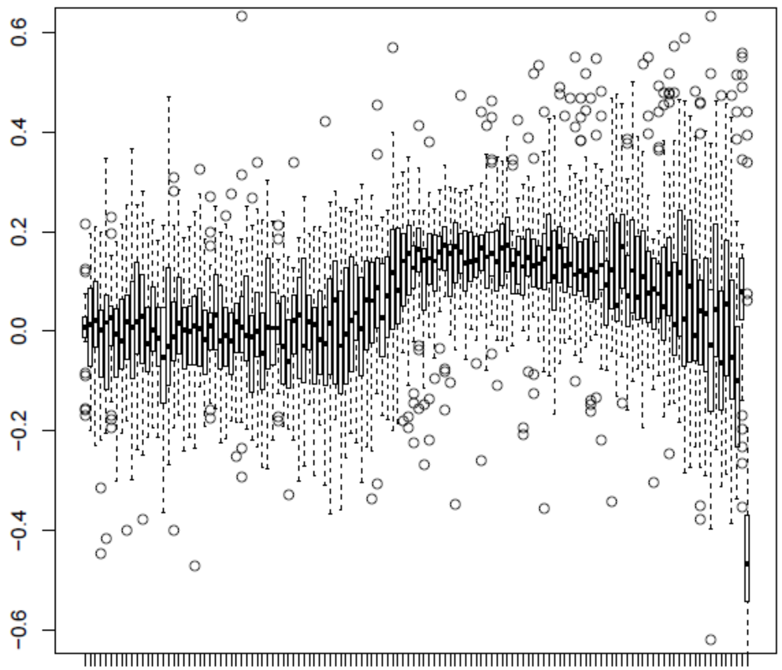
- This works well for some problems, but for others there is not enough information in a randomly generated population
- Need some convergence (c.f. WCCI 2008 paper on selection[1])
- Here running a GA to cause convergence so it is independent of model

[1]Brownlee, A. E. I., McCall, J. A. W., Zhang, Q. & Brown, D. (2008). Approaches to Selection and their Effect on Fitness Modelling in an Estimation of Distribution Algorithm, Proc. of the World Congress on Computational Intelligence 2008, Hong Kong, China, pp. 2621-2628. IEEE Press

Leading Ones

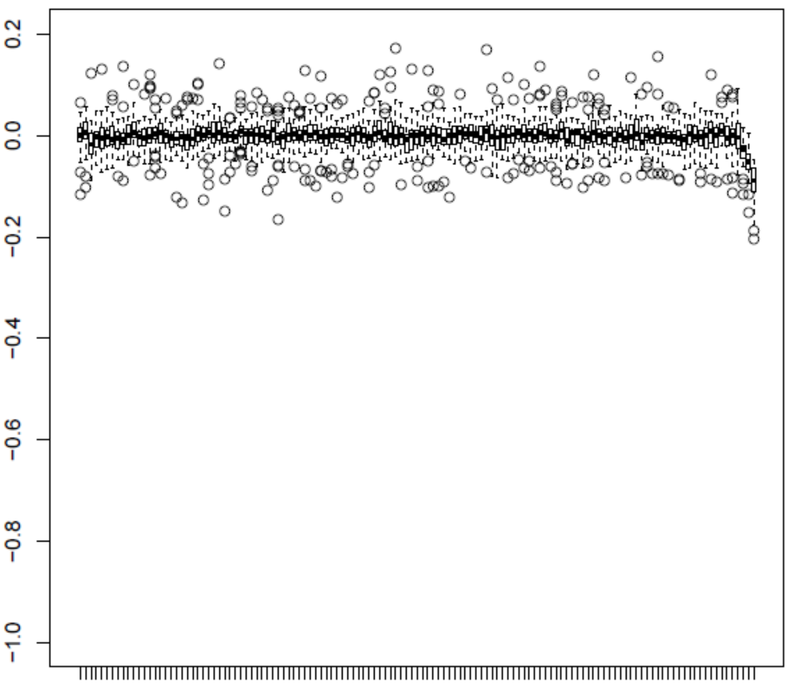
- Fitness is the count of contiguous 1s starting with x_0 in the bit string
- Univariate terms: generation 1, generation 30

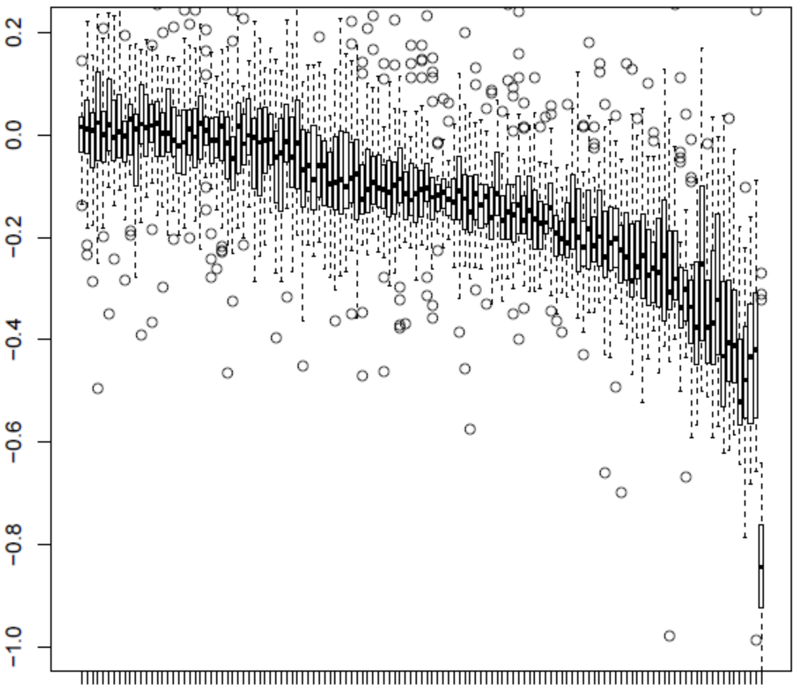




Leading Ones

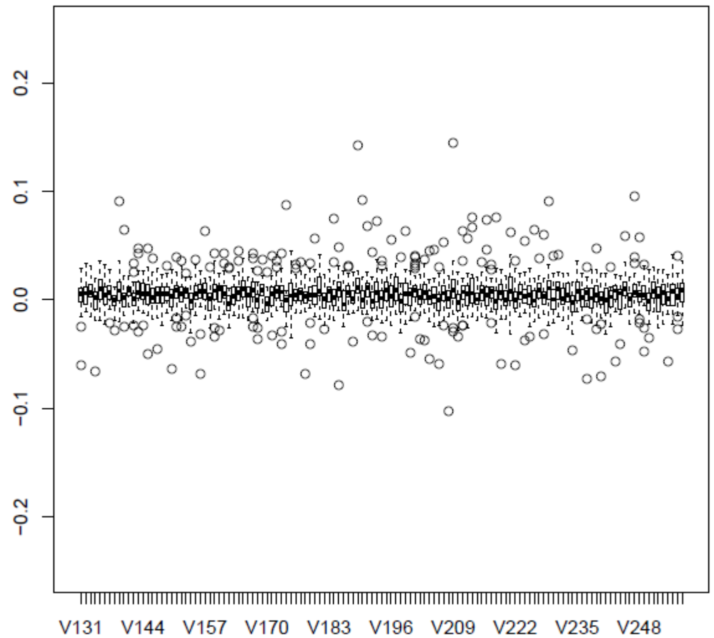
- Bivariates: terms representing neighbours in the bit string chain

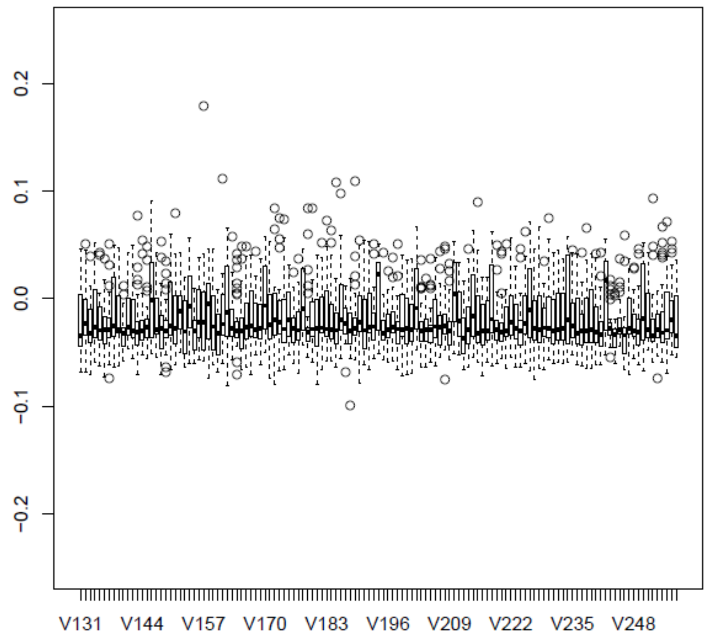




Hierarchical IF-and-only-IF

- Recursively combine blocks to get fitness: fitness gained for equal left/right halves of blocks
- Univariates: noise; Bivariates tend to -ve
- Left is generation 1, right is generation 100





Discussion

- Signs of global optima can appear very early in evolutionary process
- Often these become stronger as evolution proceeds (what we'd expect)
- Provides guidance to most sensitive variables and linkages

Adding value

- Mining the model…
  - Provides some reasoning for *why* a particular solution is optimal
  - Highlights errors in the problem definition, such as poorly defined objectives
  - Allows decision maker to choose optimal solutions wrt abstract objectives, e.g. aesthetic considerations absent from model
  - Helps identify "hitch-hiker" values

Conclusions

- When using an MBEA, we have explicit models of the fitness function
- These can be mined to gain greater insights into the problem, (almost) for free so it doesn't hurt to at least consider: "adding value" to optimisation
- How can we generalise? How might this extend to other model types?