

A “Futurist” approach to dynamic environments

Jano van Hemert, Leiden University, The Netherlands

ℰ

Clarissa Van Hoyweghen, University of Antwerp, Belgium

ℰ

Eduard Lukschandl, Ericsson & Hewlett-Packard

ℰ

Katja Verbeeck, University of Brussels, Belgium

presented by

Jano van Hemert

`jvhemert@liacs.nl`

`http://www.liacs.nl/~jvhemert`

How it all started

Coil Summer School 2000, Limerick, Ireland

- ☞ People assigned to groups to solve different problems
- ☞ Conor Ryan provided our group with two dynamic problems
- ☞ He has attempted to solve those problems using diploid chromosomes
- ☞ Our objective set was to try to solve them using one of the techniques presented at the summer school

How it all started

Coil Summer School 2000, Limerick, Ireland



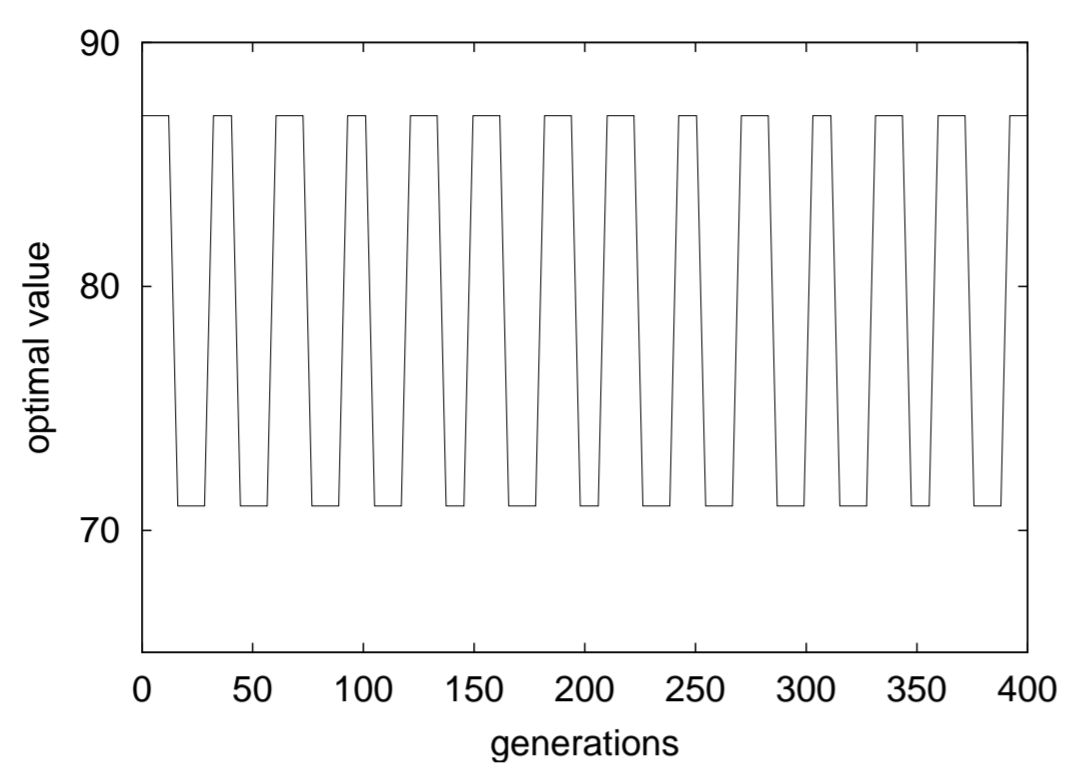
The next half hour

- ① Problem descriptions
- ② General idea
- ③ Two tested implementation
- ④ Experiments & Results
- ⑤ Conclusions & Future Work
- ⑥ Questions & Discussion

0 – 1 Knapsack – Definition

- ✓ Goal is to fill a knapsack with objects
- ✓ Each object has a weight and value assigned
- ✓ Every 15 generations the maximum allowed weight is changed
- ✓ Maximum weight is switched between 50% and 80% of the total weight of all the objects
- ✓ Total of 400 generations (time steps) is used

0 – 1 Knapsack – Behaviour



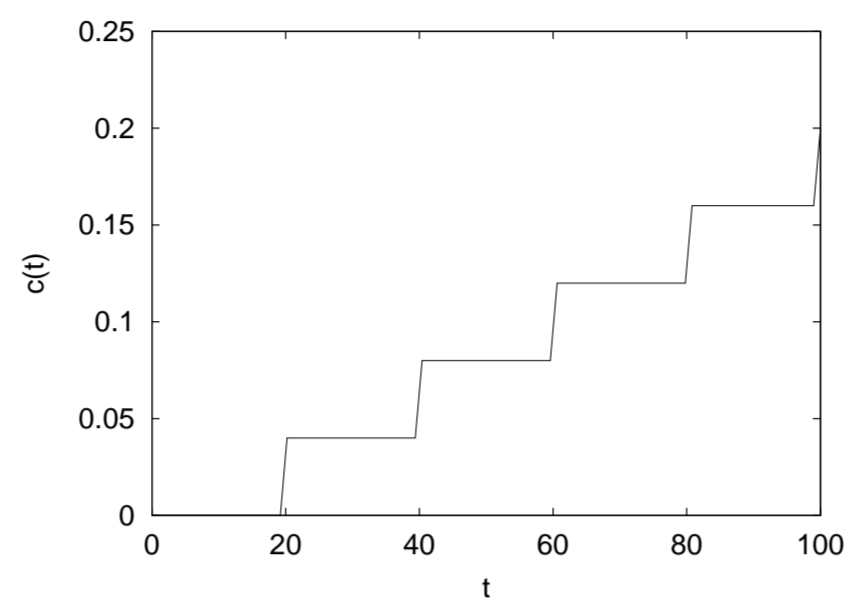
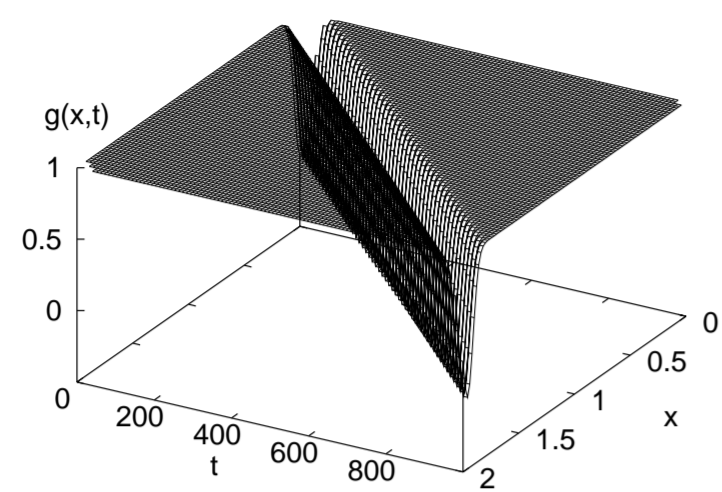
➡ Optimum changes over time

Ošmera's function — Definition

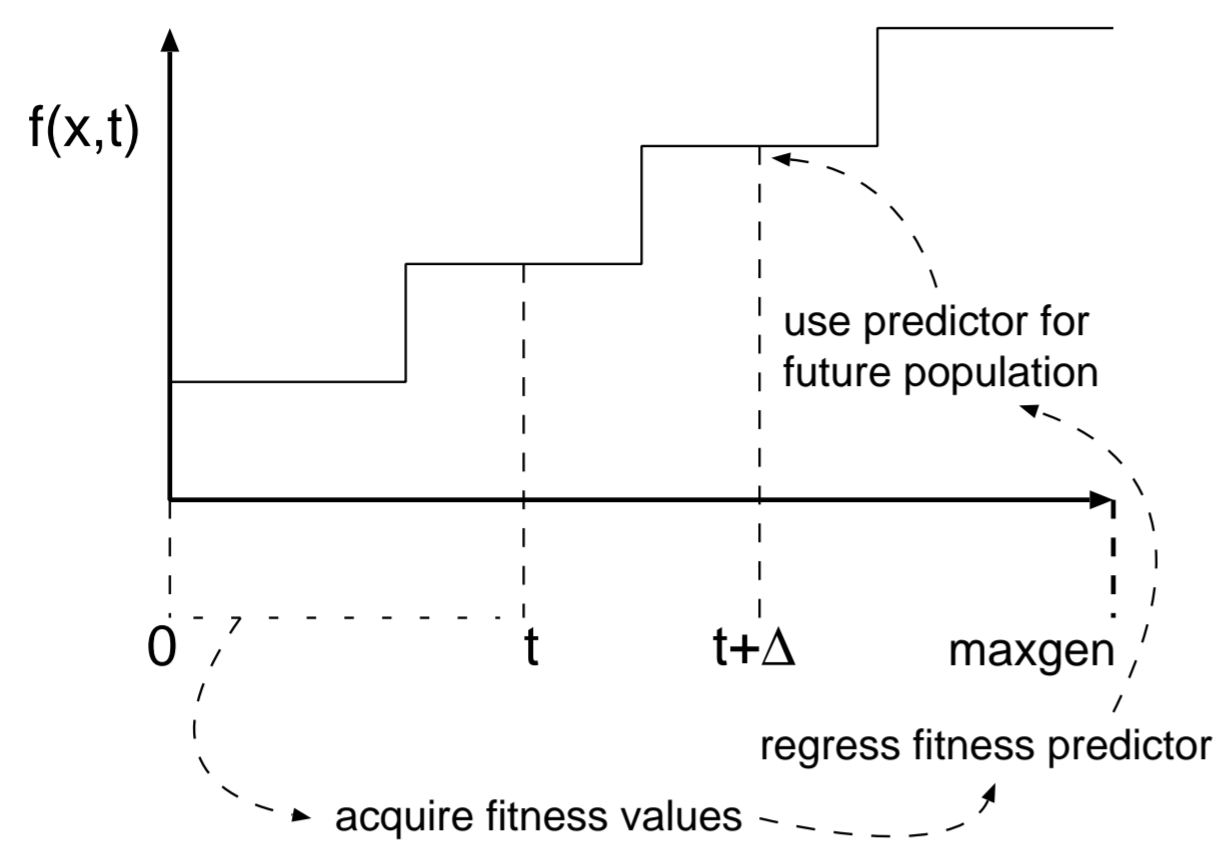
$$g_1(x, t) = 1 - e^{200(x-c(t))^2}$$

with $c(t) = 0.04(\lfloor t/20 \rfloor)$,
 $x \in \{0.000, \dots, 2.000\}$,
each time step $t \in \{0, \dots, 1000\}$ equal to one generation

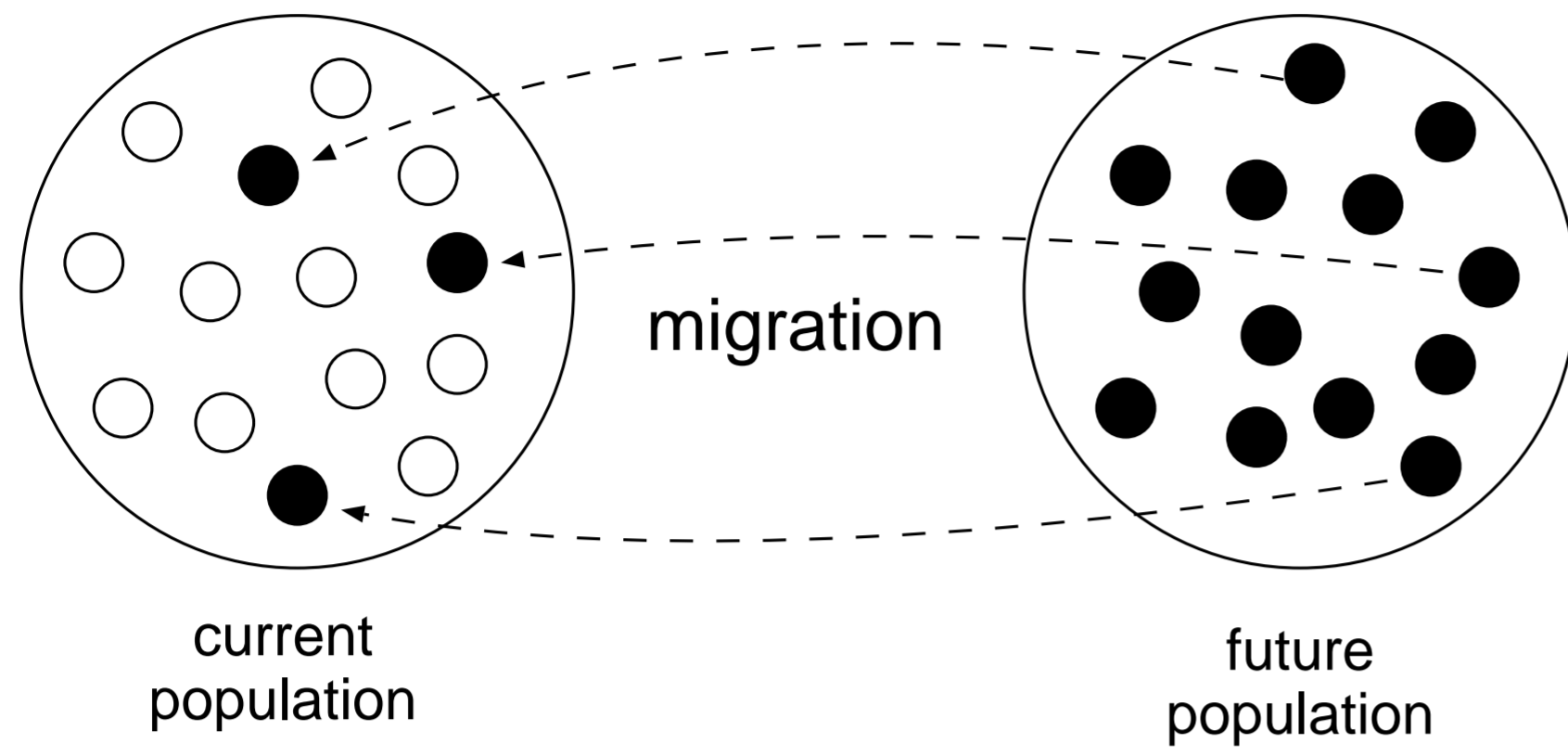
Ošmera's function — Behaviour



Predicting the future



Learning from the future



Parameters

✓ m determines how many individuals are copied to the current population, best m from the future are selected and overwrite the worst m in the current population

✓ Δ determines how many generations ahead the future population lives

Two experiments

Perfect prediction

- ☞ Idea is that the best what could happen is that you have a perfect prediction of the future
- ☞ With these problems this is very easy to implement as we know exactly the optimum for $t + \Delta$
- ☞ If this is not successful, we could ask ourselves if it is useful to continue with the idea of predicting the future

Noisy prediction

- ☞ Could the use of a predictor be harmful?
- ☞ We give the algorithm noisy and deceptive predictions of the future
- ☞ Knapsack problem gets wrong optimum (deceptive) and Ošmera gets a random value

Experimental setup

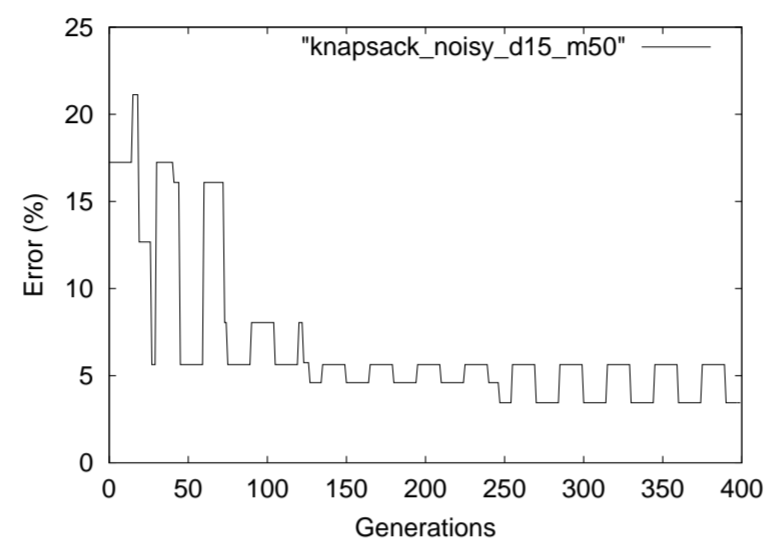
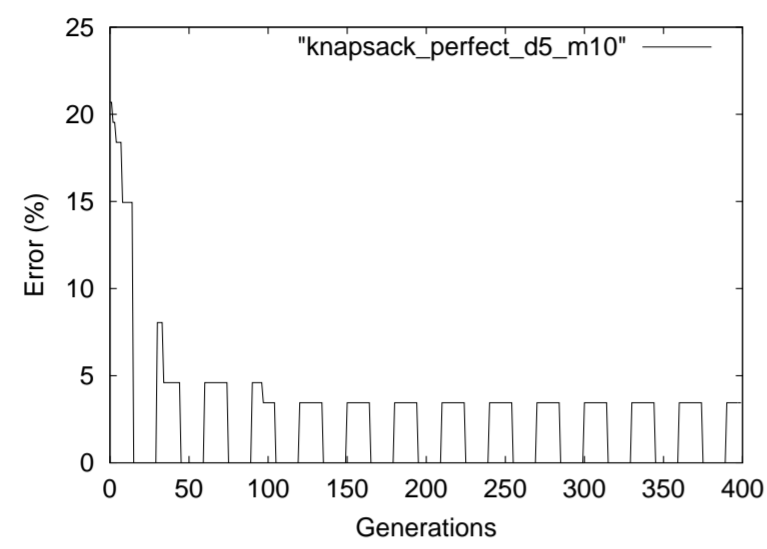
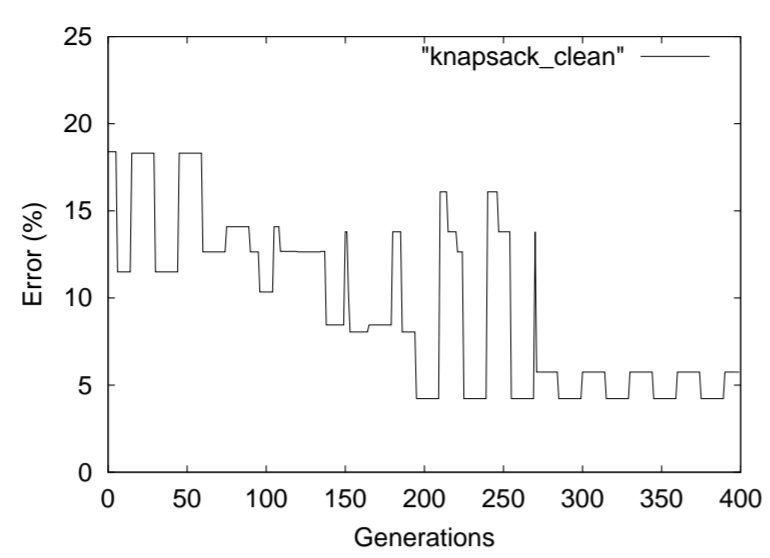
For both problems we do

- ✓ a test without any future population
- ✓ tests with four parameter settings (two pairs) for perfect prediction
- ✓ tests with four parameter settings (two pairs) for noisy / deceptive predictor
- ✓ 50 runs for each test with unique random seeds

Knapsack results

<i>predictor</i>	Δ	<i>m</i>	<i>error</i>	<i>stdev</i>	<i>best run</i>
none	×	×	16.6%	3.52	8.96%
perfect	5	10	11.9%	3.77	4.85%
perfect	15	10	20.3%	4.26	11.7%
perfect	5	50	11.8%	3.70	5.97%
perfect	15	50	21.4%	6.06	12.0%
deceptive	5	10	12.6%	4.12	6.77%
deceptive	15	10	13.0%	3.74	7.50%
deceptive	5	50	12.7%	4.02	6.47%
deceptive	15	50	12.8%	4.07	4.99%

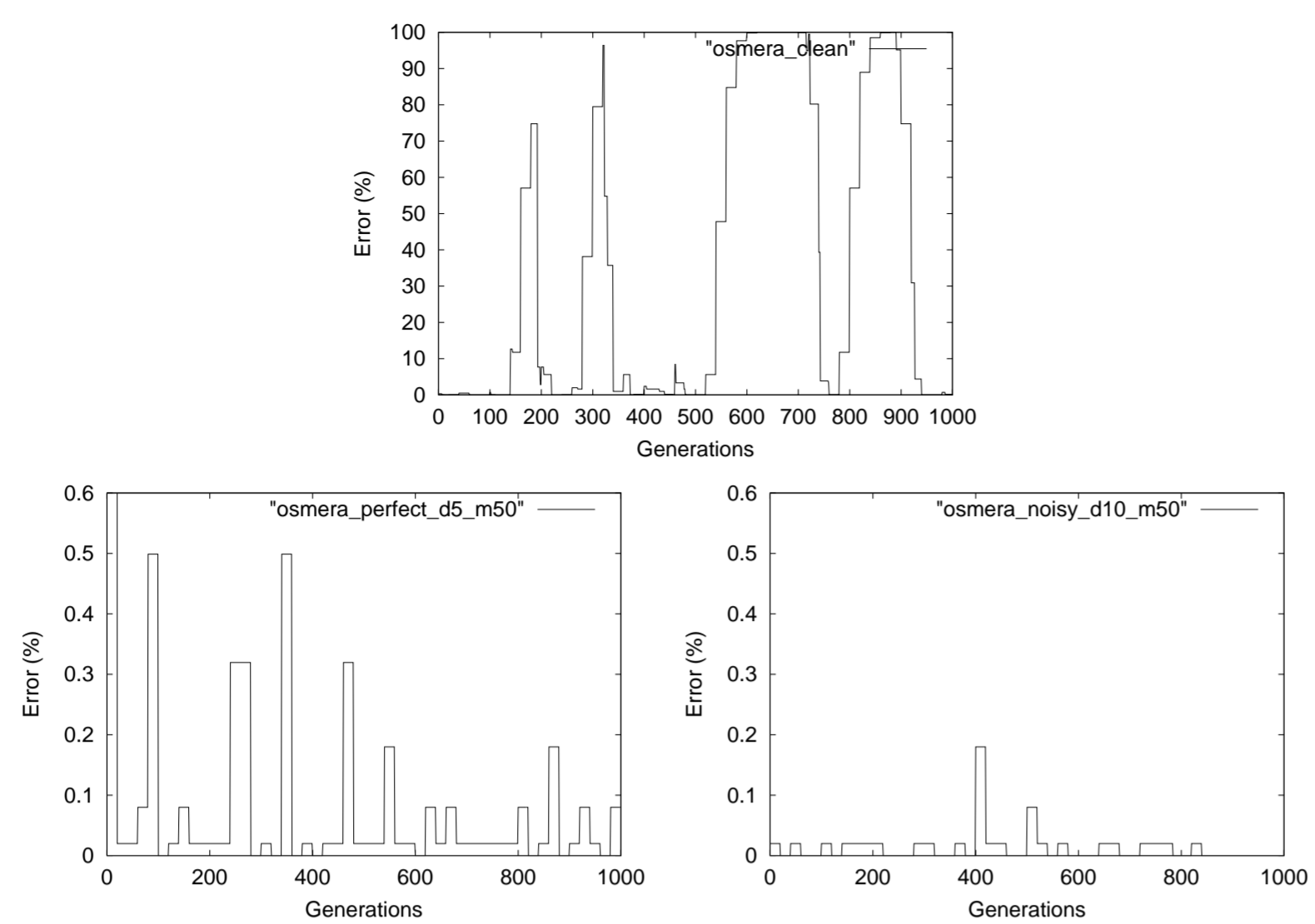
Knapsack results



Ošmera results

<i>predictor</i>	Δ	<i>m</i>	<i>error</i>	<i>stdev</i>	<i>best run</i>
none	×	×	63.8%	10.3	41.2%
perfect	5	10	0.261%	0.153	0.0751%
perfect	5	50	0.168%	0.148	0.0266%
perfect	10	10	0.241%	0.220	0.0680%
perfect	10	50	0.203%	0.099	0.0698%
noisy	5	10	0.241%	0.186	0.0488%
noisy	5	50	0.144%	0.122	0.0358%
noisy	10	10	0.241%	0.186	0.0488%
noisy	10	50	0.168%	0.148	0.0266%

Ošmera results



Conclusions

Pros and cons

- ✘ Knapsack problem is better solved with a look-a-head time of 5 generations as opposed to 15, which is the length of the cycle
- ✓ Adding future predictions when solving the knapsack problem slightly improves the performance when using a deceptive function or when using small values for m
- ✓ Adding future predictions when solving Ošmera's function seems to help
- ✘ There is little difference in performance between using a perfect or noisy predictor...
- ✓ There is little difference in performance between using a perfect or noisy predictor...

Future Research

- ⇒ How sensitive are the parameters m and Δ ?
- ⇒ Why does this work well for a real-valued problem and not for a problem from a discrete domain?
- ⇒ Could we replace this whole complicated process by adding more disturbance? For instance with a high mutation rate?