Stochastic Search Heuristics in Finance

Thiemo Krink

Group Investments, Allianz Investment Management (AIM), Munich, Germany

Sandra Paterlini

Dept. of Economics,CEFIN and RECent, Univ.of Modena and Reggio E., Italy

Introduction

Optimization in finance

- Portfolio optimization
- Index tracking / benchmark replication
- Credit rating (PD bucketing)
- Asset model calibration
- Fund style analysis
- Time-series prediction
- Trading rule discovery
- Arbitrage discovery
- ...

Introduction

How can we tackle these problems?

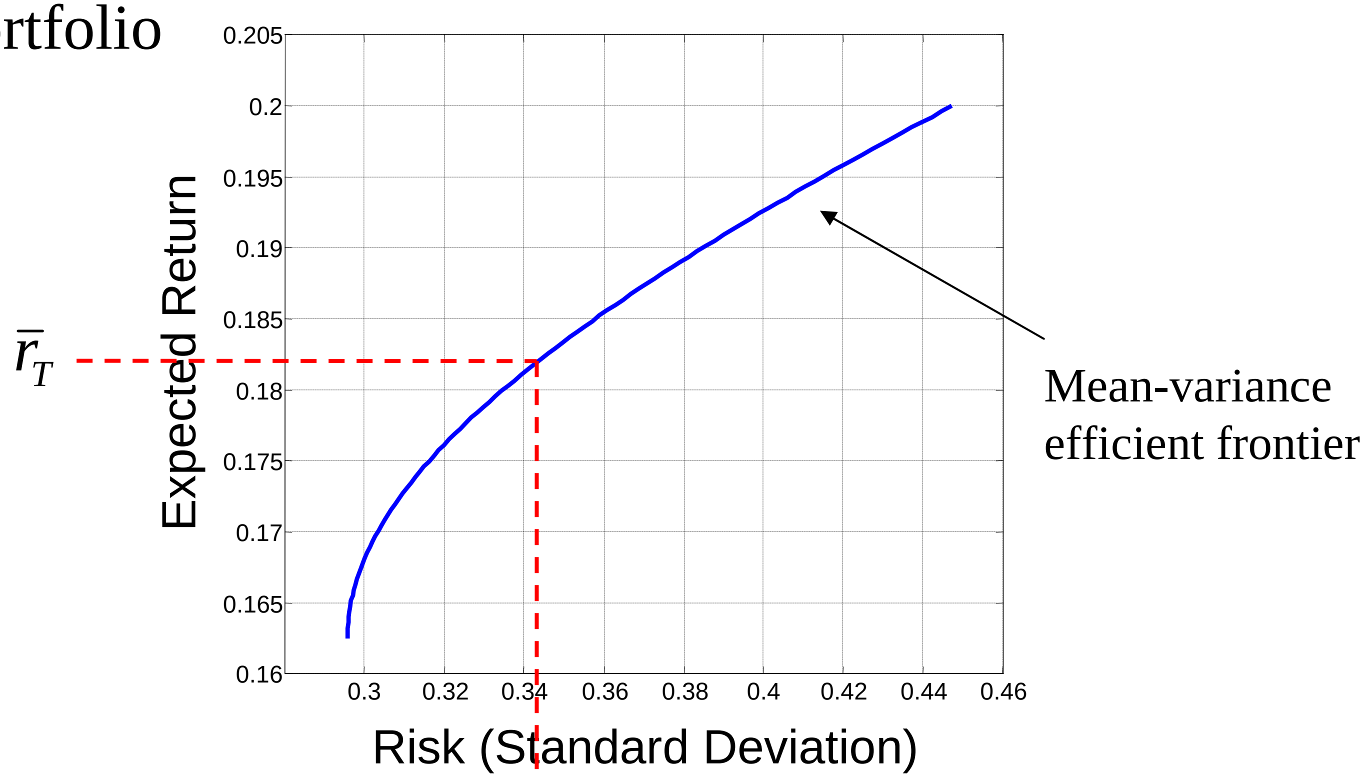
- Closed-form solution
- Linear or quadratic programming (LP/QP)
- Partial quadratic prog. combined with dynamic programming
- Stochastic search heuristics

Portfolio Optimization with LP/QP

Objective

(Markowitz 1952)

- Calculate the mean-variance efficient frontier for a portfolio



$$\sigma^2 = \mathbf{w}^T \Sigma \, \mathbf{w}$$

Portfolio Optimization with LP/QP

Calculation of the efficient frontier (Step 1)

- calculate the the extreme points of the expected return

$$\min \quad \mathbf{w}^T \boldsymbol{\Sigma}\, \mathbf{w}$$

$$\sum_{i=1}^{n} w_i = 1$$

$$w_i \geq 0$$

QP problem

$$\max \quad \mathbf{w}^T \bar{\mathbf{r}}$$

$$\sum_{i=1}^{n} w_i = 1$$

$$w_i \geq 0$$

LP problem

minimum risk portfolio
=> minimum return portfolio $\bar{r}_{\min}$

maximum return portfolio $\bar{r}_{\max}$

Portfolio Optimization with LP/QP

Calculation of the efficient frontier (Step 2)

- determine the intermediate points of the curve by calculating the minimal risk for n intermediate returns $\bar{r}_T$ between the extreme points $\bar{r}_{\min}$ and $\bar{r}_{\max}$

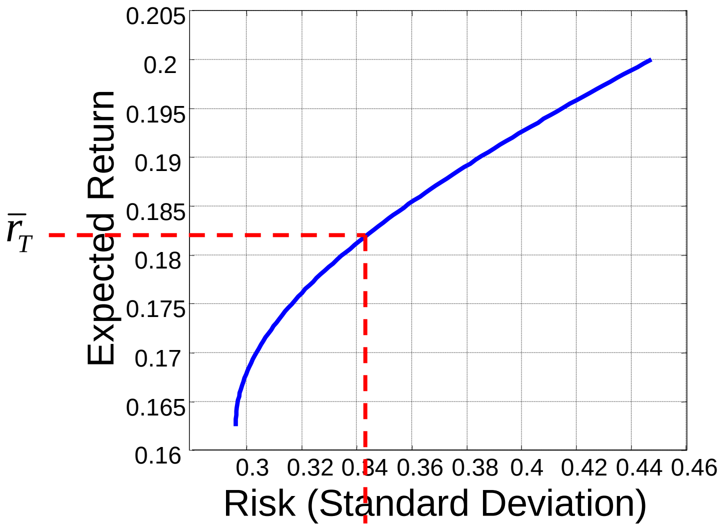
$$\min \quad \mathbf{w}^T \boldsymbol{\Sigma} \, \mathbf{w}$$

$$\text{s.t.} \quad \mathbf{w}\bar{\mathbf{r}} = \bar{r}_T$$

$$\sum_{i=1}^{n} w_i = 1$$

$$w_i \geq 0$$

another QP problem



$$\sigma^2 = \mathbf{w}^T \boldsymbol{\Sigma} \, \mathbf{w}$$

Introduction

Why do we need stochastic search?

- Conventional techniques require rigid assumptions (convexity, linearity, differentiability, explicitly defined objectives, problem can be split into subproblems, etc.)
- In most applications, the only available information regarding the objective function, is its value
- Often the objective function or constraints are discontinuous

Introduction

Why do we need stochastic search (cont.)?

- Often we get away with simplifications (linearization, convexity assumptions, etc.), but not in all cases!
- Conventional techniques lack generality; new applications often require from scratch implementation
- Stochastic search allows to obtain results anytime prior to termination

Portfolio Optimization beyond LP/QP

A more realistic case...

- Non-trivial risk and value objectives (VaR, Expected Shortfall, utility functions, no normality assumpt.)
- Cardinality constraints (e.g.: an upper bound for the number of assets in the portfolio, asset selection)
- Buy-in thresholds (e.g.: an asset can be included only if its amount is bigger/smaller than a lower/upper limit)
- Roundlots (e.g.: the smallest volume of an asset that can be bought)
- …

Introduction

(Fogel, 1966)
(Rechenberg, 1973)
(Holland, 1975)

Evolutionary Computation as an example

- Evolution can be considered as an optimization process, which is very general and can deal with highly complex problems
- Darwinian evolution and Mendelian inheritance can serve as a model for optimization algorithms

Introduction

Evolutionary Computation as an example

- Describe a problem by a set of parameters
- Interpret the parameters as artificial genes
- Consider the genes as blueprints of individuals
- Apply evolution to the individuals (survival of the fittest)

The algorithm

Problem

quality measure: **f** (x_1, x_2, x_3, x_4)

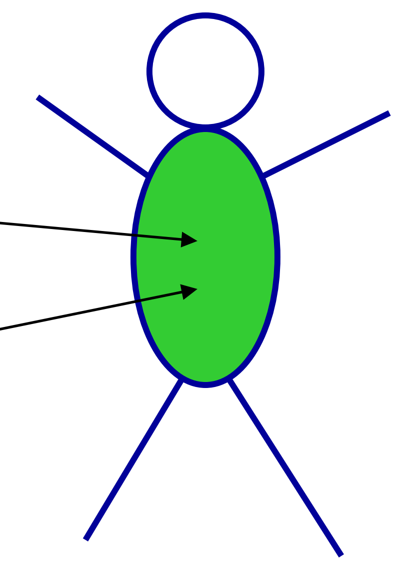
problem parameters: $\mathbf{x_1}$, $\mathbf{x_2}$, $\mathbf{x_3}$, $\mathbf{x_4}$

artificial **genes**: | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |

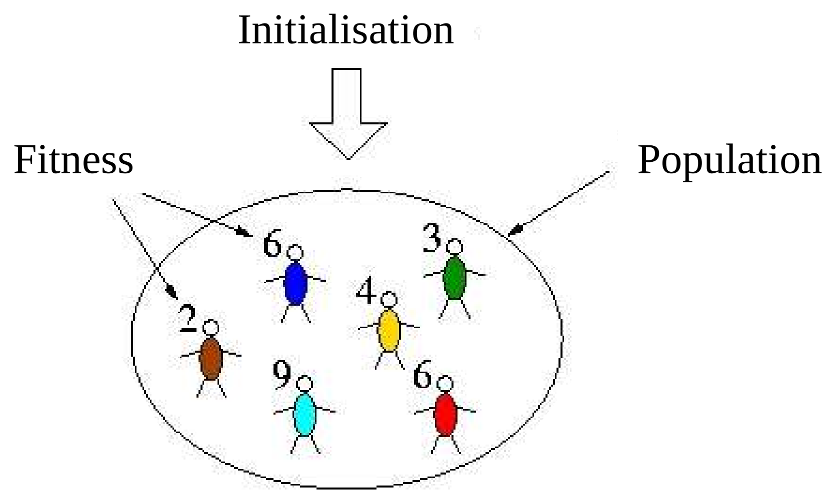
representation of one solution

fitness: **f** (**genes**)

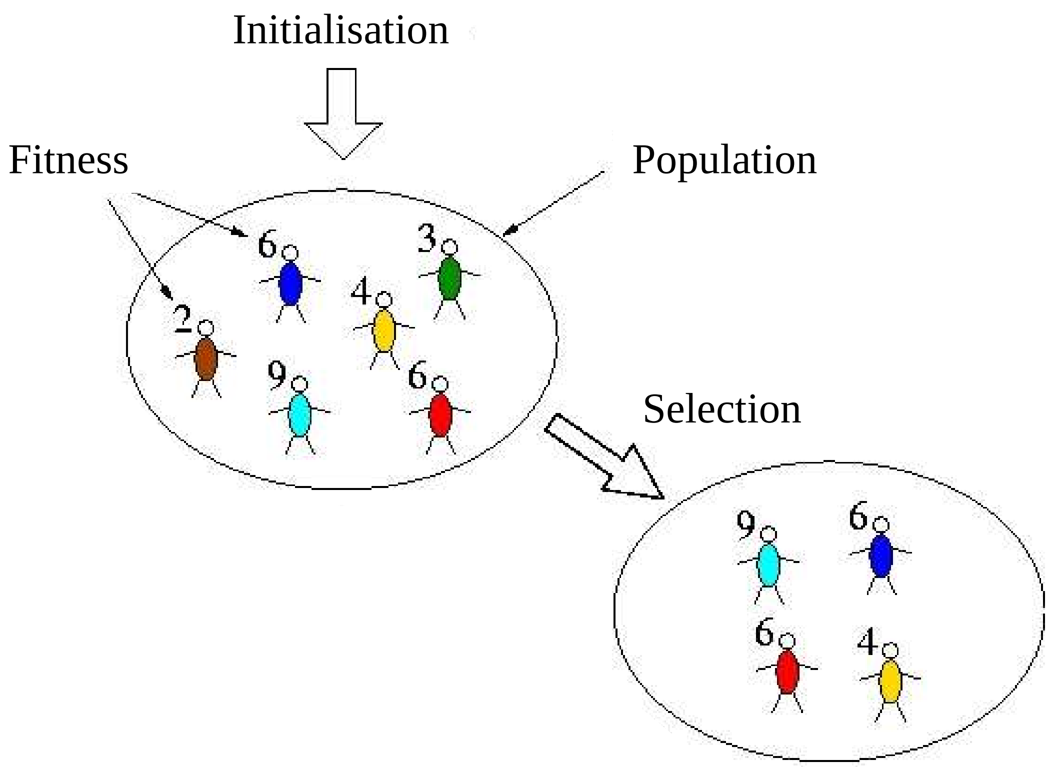
Individual



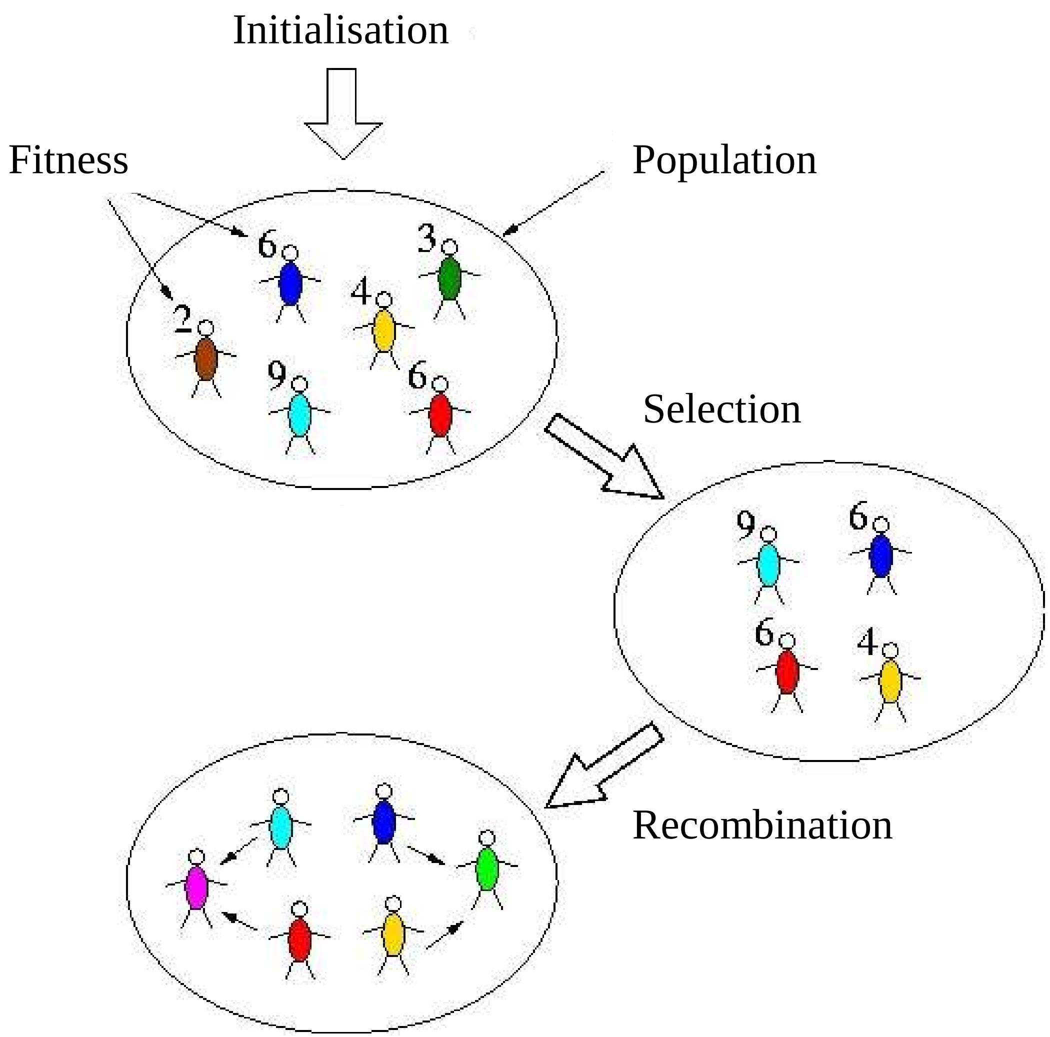
The algorithm



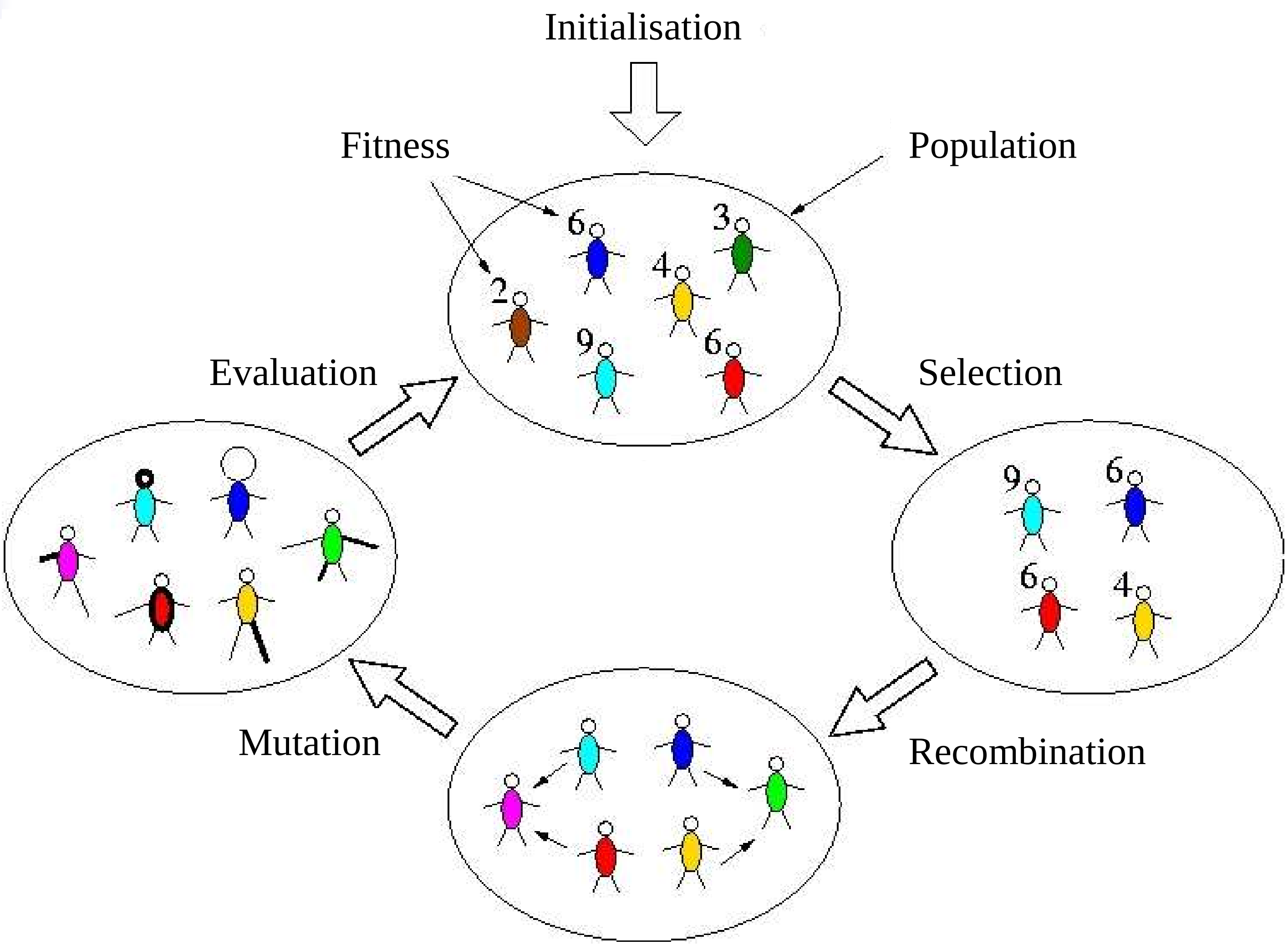
The algorithm



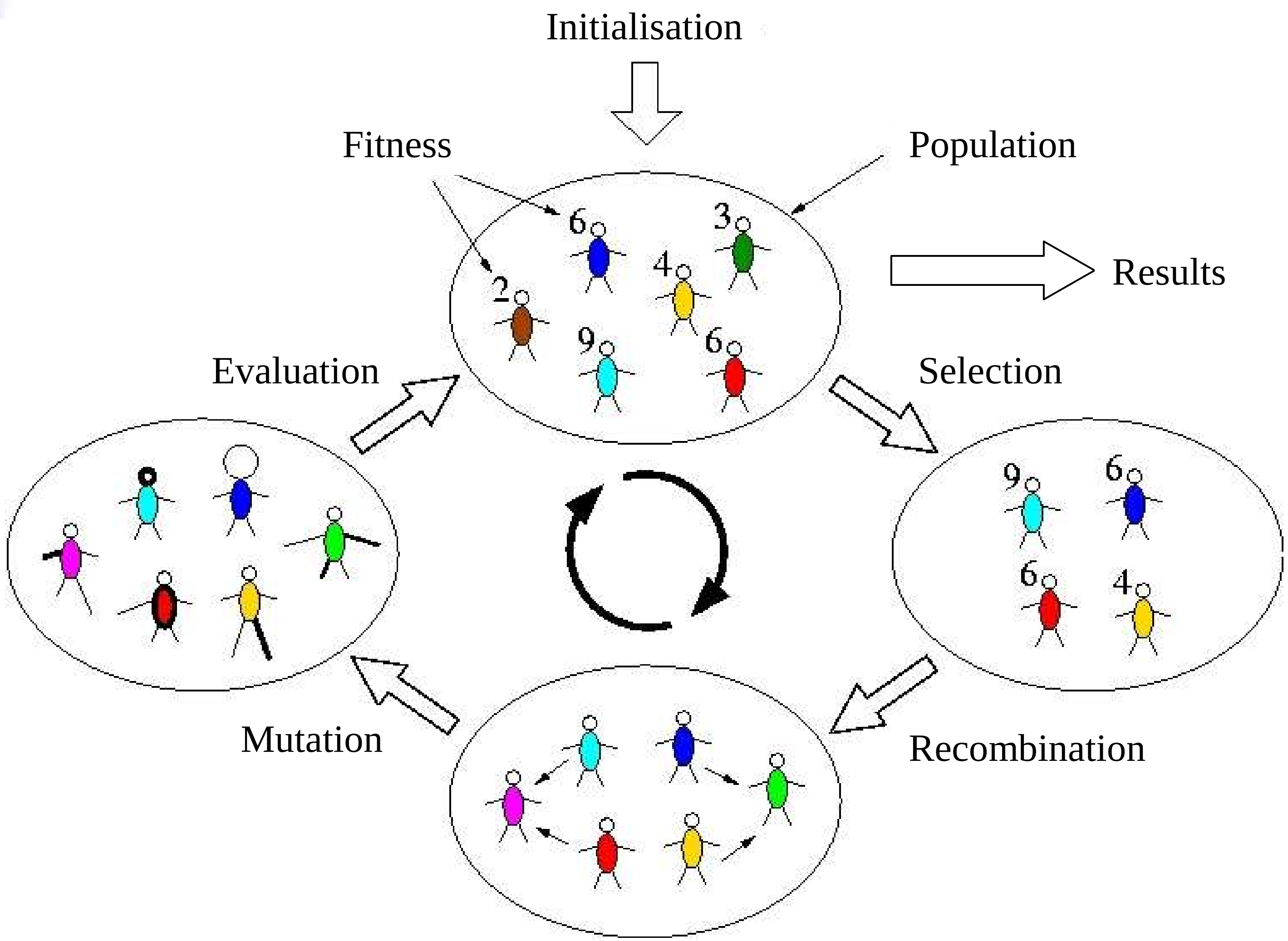
The algorithm



The algorithm



The algorithm



Pseudo-code

```
void EvolutionaryAlgorithm()
{
    t = 0;
    initialise population P(t);
    evaluate population P(t);   // calculate fitnesses
    while (not termination condition) {
        t = t + 1;
        select next generation P(t) from P(t-1);
        alter P(t);   // mutate and recombine genes
        evaluate population P(t);   // calculate fitnesses
    }
}
```

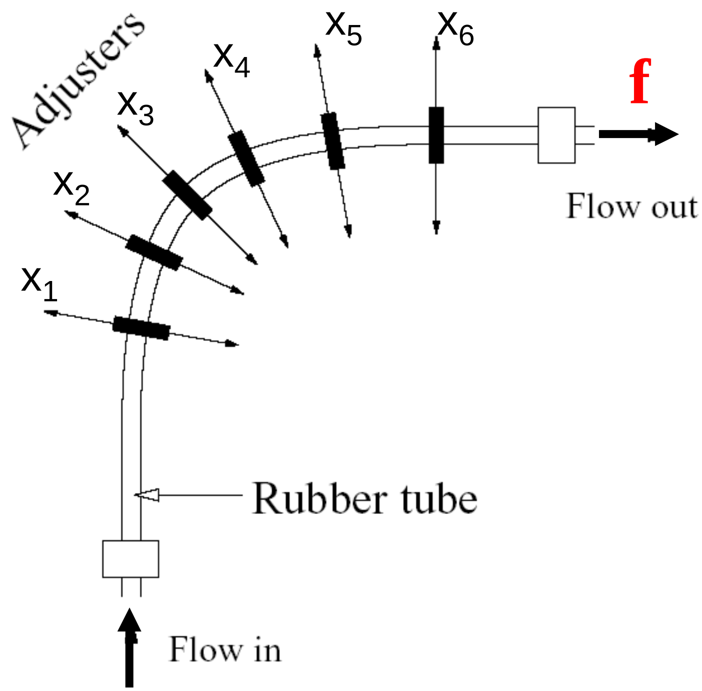
P(t) = Population at time t

A case study

(Rechenberg, 1973)

The task: Design a bent tube with a maximum flow

Goal: gas flow $\mathbf{f}\,(x_1, x_2, \ldots, x_6) = \mathbf{f}_{\mathbf{max}}$

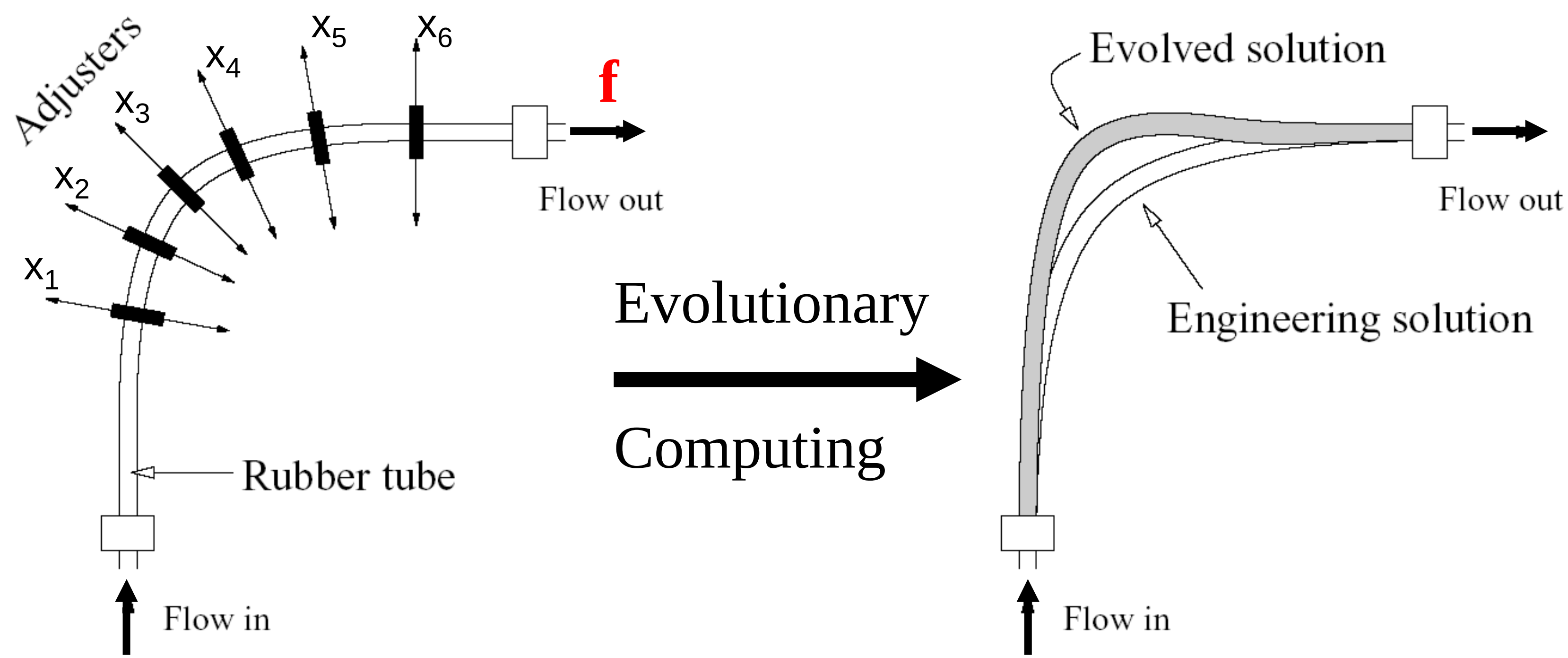


A case study

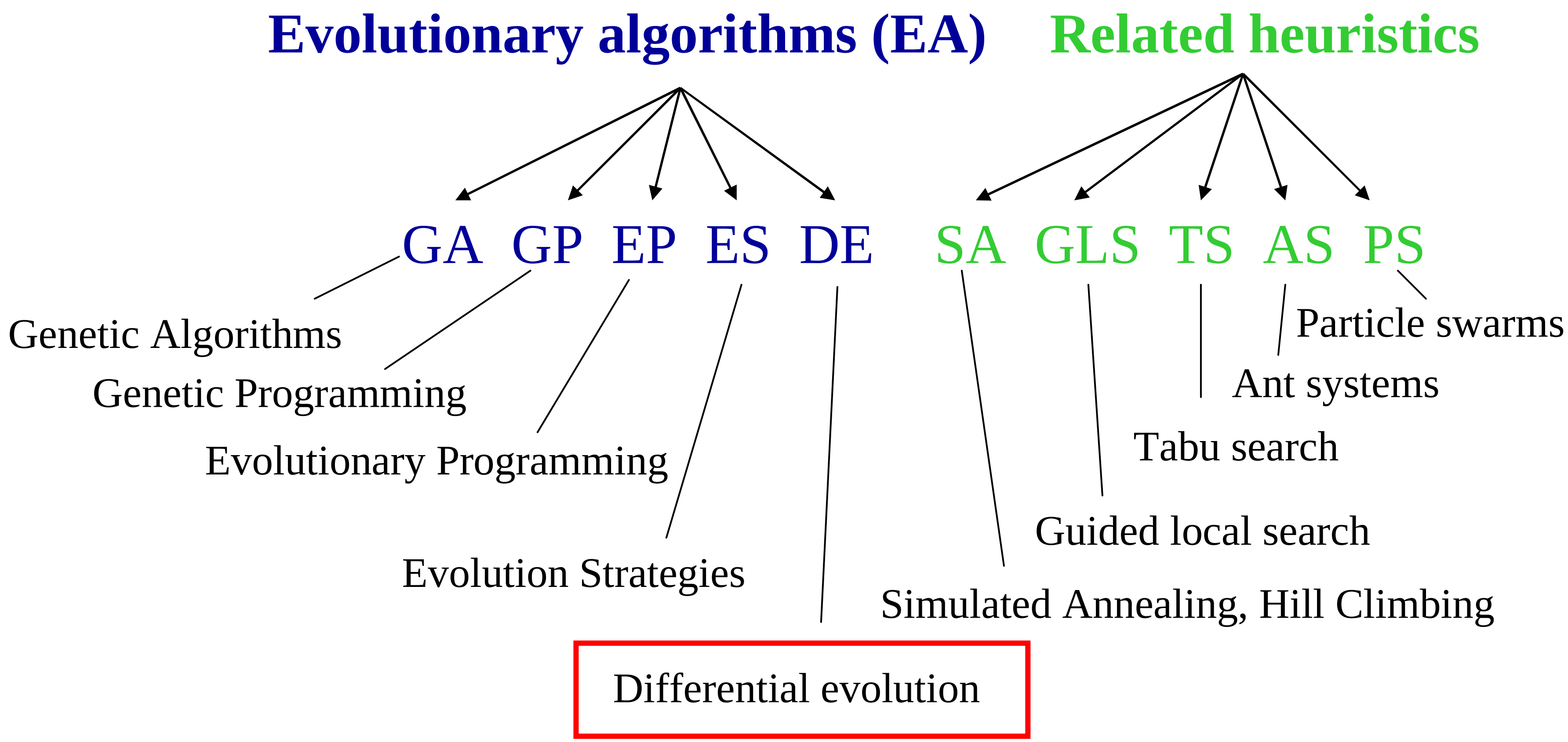
(Rechenberg, 1973)

The task: Design a bent tube with a maximum flow

Goal: gas flow $\mathbf{f}$ $(x_1, x_2, \ldots, x_6) = \mathbf{f}_{max}$



EAs and related search heuristics



Differential Evolution

Application

- continuous numerical optimization

Properties

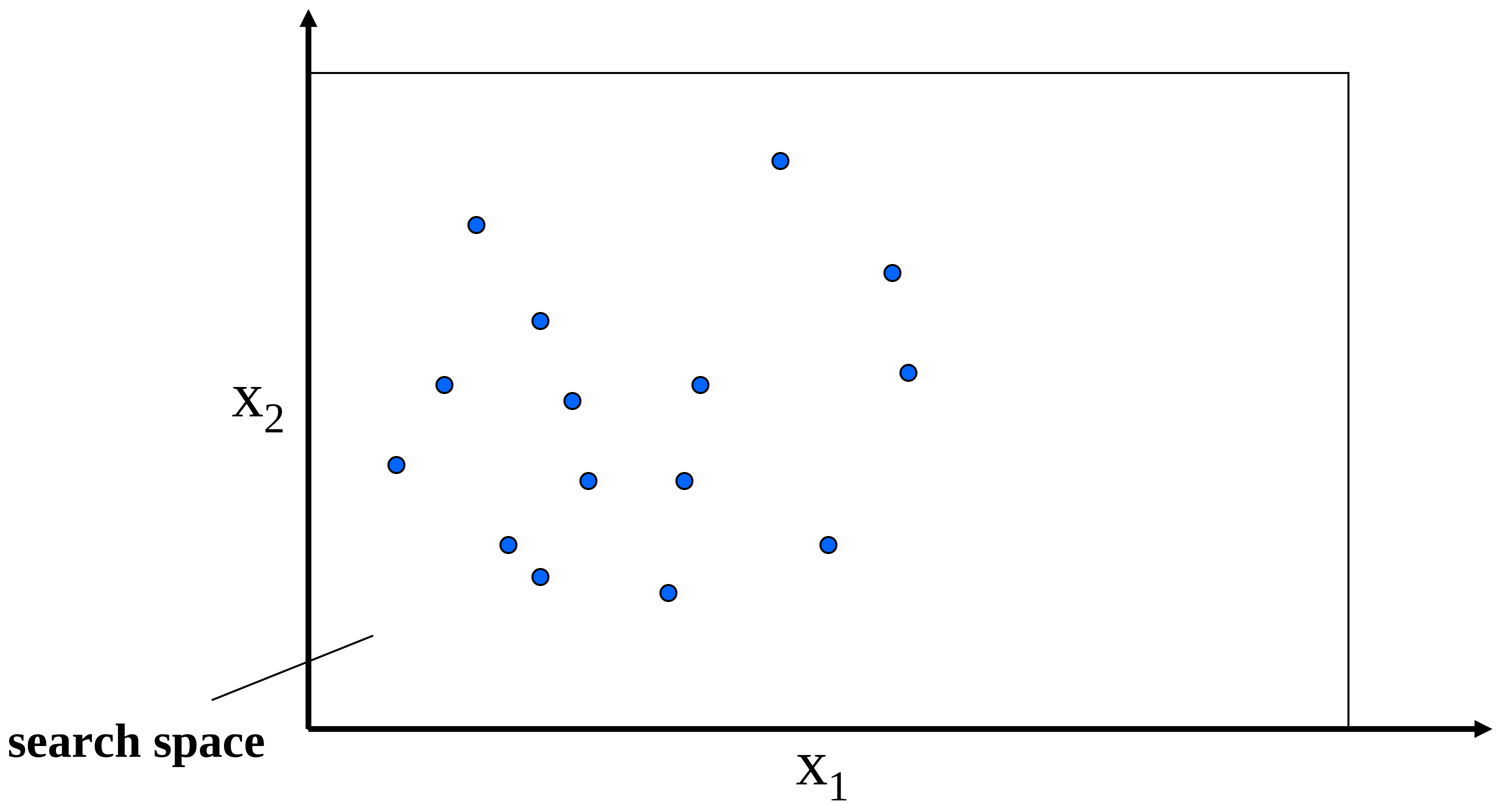
- high accuracy
- robustness (no occasional bad runs)
- works very well for various kinds of real-world problems
- same parameter settings work fine for a wide range of problems (no need for tuning)
- easy to implement
- drawback: somewhat slow during the first iterations

Differential Evolution

(Storn and Price, 1995)

Step 1 - Initialize and evaluate

- Generate a random start population and evaluate the individuals

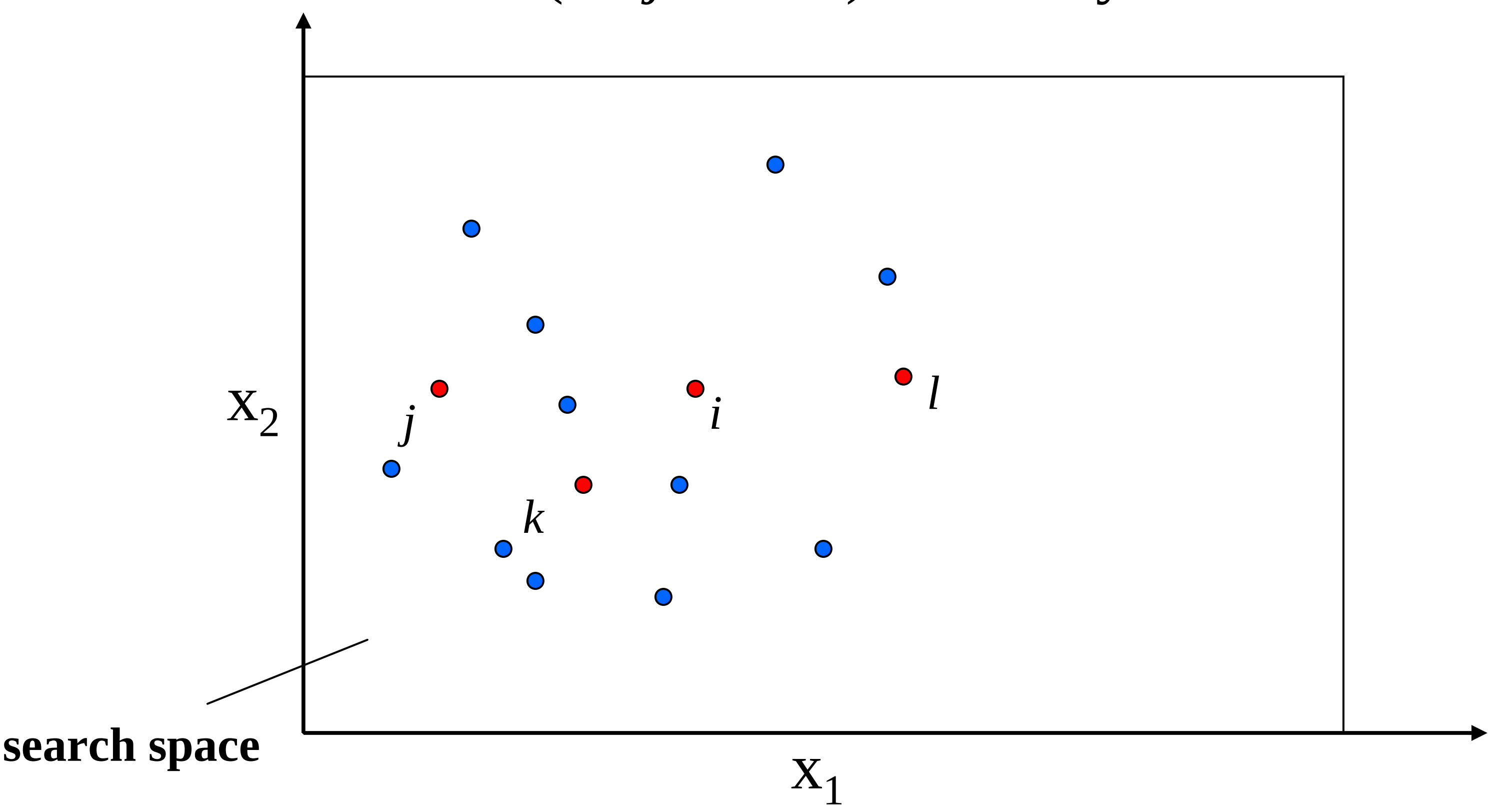


Differential Evolution

(Storn and Price, 1995)

Step 2 - Evolve (repeat until termination)

- For each individual *i*, select three other individuals *j*, *k* and *l* with ($i \neq j \neq k \neq l$) randomly

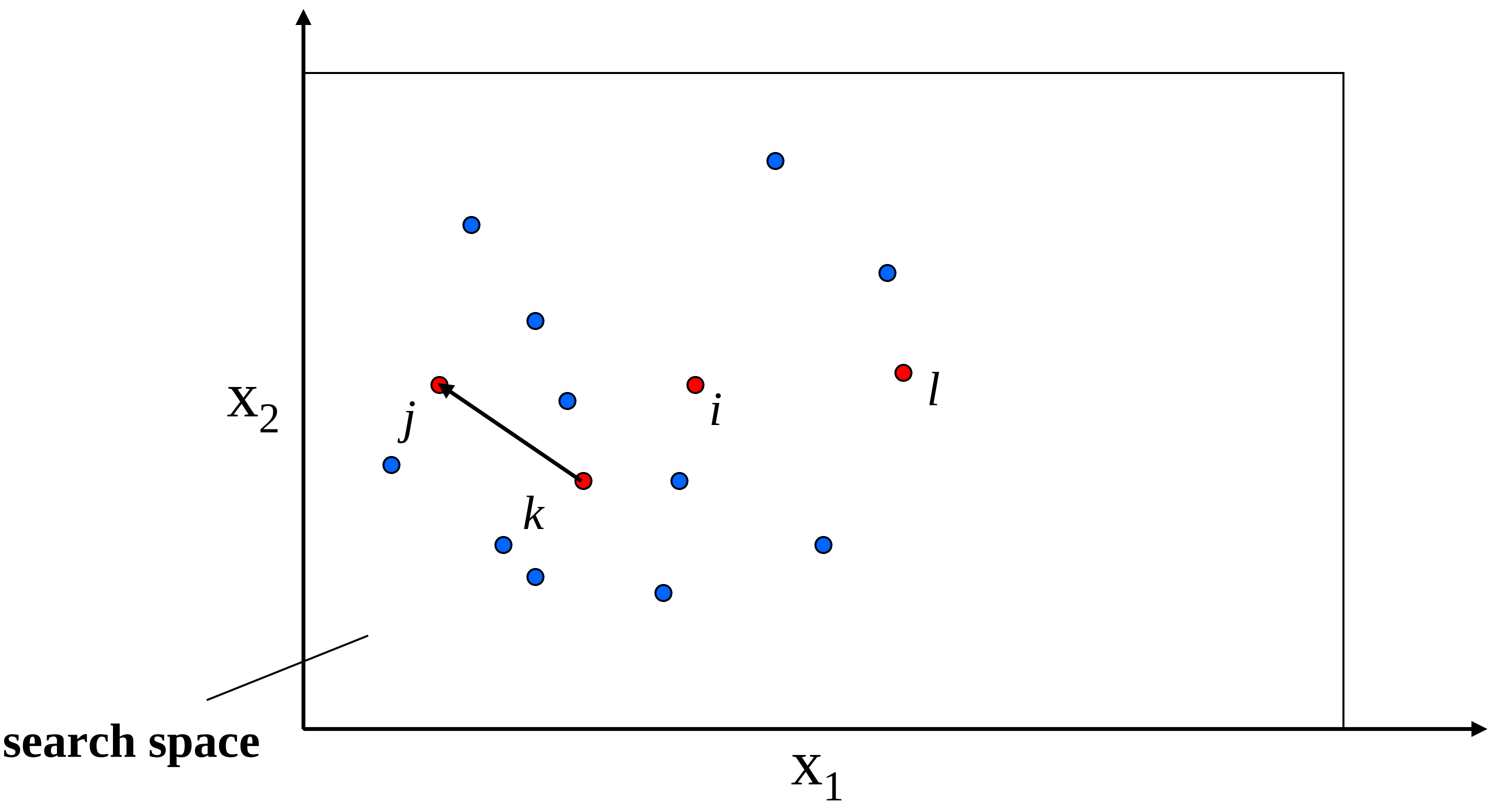


Differential Evolution

(Storn and Price, 1995)

Step 3

- Calculate the difference vector between *j* and *k*

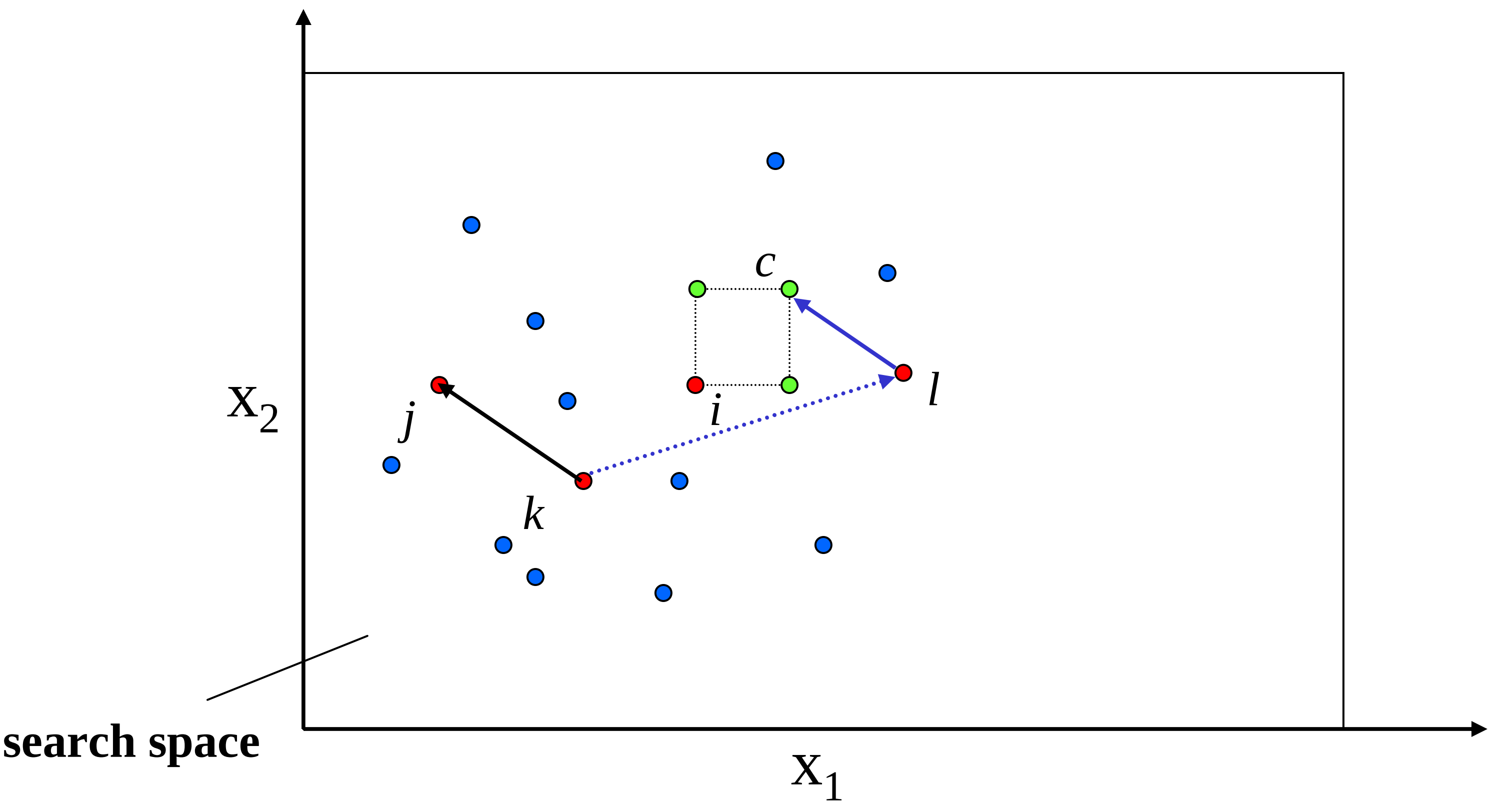


Differential Evolution

(Storn and Price, 1995)

Step 4

- Add the scaled difference vector to *l* to create a candidate *c* as a crossover of this result and individual *i*

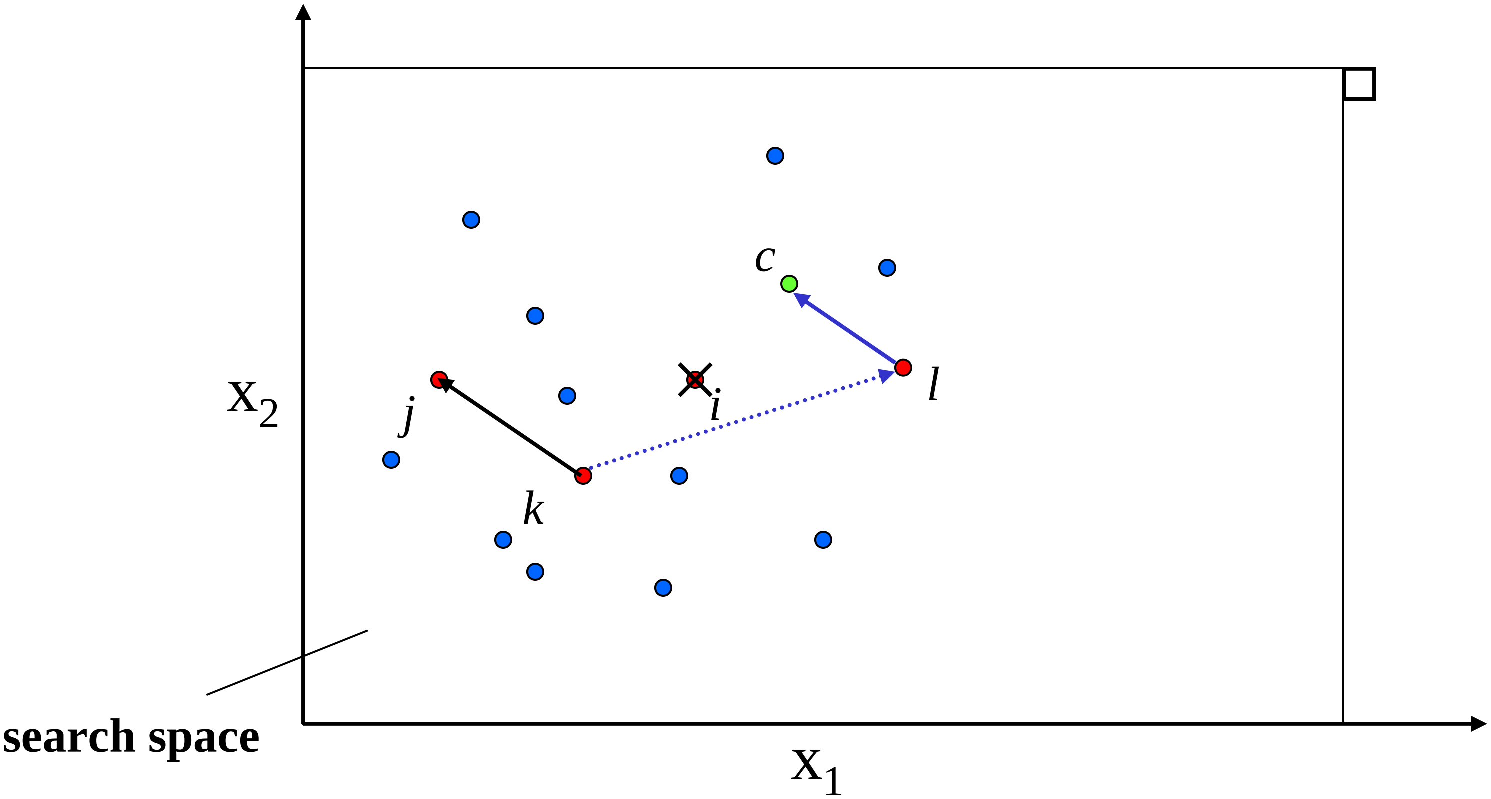


Differential Evolution

(Storn and Price, 1995)

Step 5

- Replace *i* with *c* if it has a better fitness and continue with step 2 (for all individuals; repeat until termination)



Index Tracking with Differential Evolution
-
An Application

The Index Tracking Problem

Application context

- Construction of an index tracking fund, such as an ETF
- Construction of a benchmark for asset managers

Goal

- Replication of an index by a subset of its underlying assets

Optimization

- Find the optimal subset of assets and their optimal allocation (asset weights), such that the distance between the tracking portfolio and the index is minimal
- Typically additional non-linear constraints have to be considered in real-world applications

Problem specification

Objective: $\underset{\mathbf{w}}{minimize} \quad f(\mathbf{w}) = distance(R^P - R^I) \qquad \mathbf{w} \in \Re^n$

continuous numerical problem

Constraints: *subject to*

$$(1) \sum_{i=1}^{n} w_i = 1$$

$$(2) \ 0 \le w_i \le 1$$

$$(3) \ \varepsilon_i \delta(w_i) \le w_i \le \xi_i \delta(w_i), \delta(w_i) = \begin{cases} 1 & if \ w_i > 0 \\ 0 & otherwise \end{cases}$$

$$(4) \ L \le \sum_{i=1}^{n} \delta(w_i) \le K$$ ← **integer constraint**

$$(5) \sum_{i: w_i > Lb} w_i \le Ub$$ ← **conditional sum constraint**

$$(6) \ \mathbf{Aw} \le \mathbf{b} \ \ and \ \ (\mathbf{A}_{eq}\mathbf{w} = \mathbf{b}_{eq})$$

$$(7) \sum_{i=1}^{n} | w_i - w_i^{old} | < Turnover$$ ← **non-linear constraint**

The Index Tracking Problem

The distance measure

- We consider two objective functions
  - if we do not know the index composition

$$\underset{\mathbf{w}}{minimize} \quad f(\mathbf{w}) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(R_t^{\,P} - R_t^{\,I})^2} \qquad (1)$$

  - if we know the index composition (QP)

$$\underset{\mathbf{w}}{minimize} \quad f(\mathbf{w}) = (\mathbf{w} - \mathbf{wb})\mathbf{\Sigma}(\mathbf{w} - \mathbf{wb})^T \qquad (2)$$

- DECS-IT can easily deal with other functions

Problem characteristics

Challenges

- Large scale optimization (100-600 dimensions)
- Integer constraint (with QP/LP requires branch and bound, but huge number of combinations to be tested!)
- Conditional sum constraint (non QP/LP compatible)
- Turnover constraints (non QP/LP compatible)

Problem characteristics

Fitness landscape

- many (!) near optimal solutions
- with limited number of positions, the position selection crucially shapes the landscape
- for fitness function $\underset{\mathbf{w}}{minimize}\ f(\mathbf{w}) = (\mathbf{w} - \mathbf{wb})\boldsymbol{\Sigma}(\mathbf{w} - \mathbf{wb})^T$ without constraints => trivial optimal solution at: **w = wb**

Choice of Methods

The DECS-IT Optimization Algorithm

1. Differential Evolution (DE)
2. Position selection
3. Constraint handling

Experimentation with In-Sample Replication

$$\underset{\mathbf{w}}{minimize} \quad f(\mathbf{w}) = (\mathbf{w} - \mathbf{wb})\mathbf{\Sigma}(\mathbf{w} - \mathbf{wb})^T$$

subject to the mentioned constraints

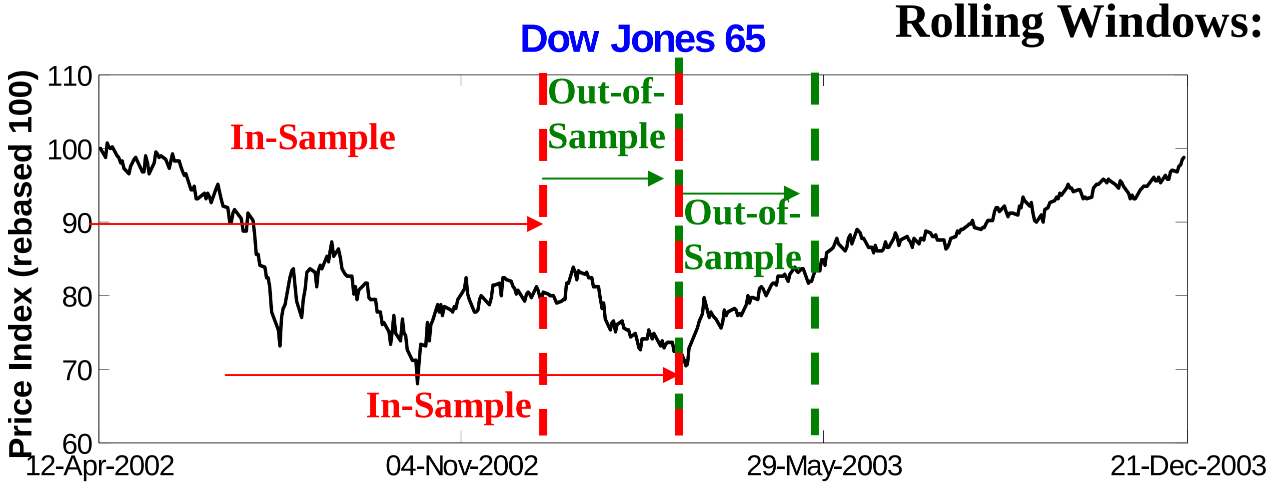
=> Live demo in Matlab

Experimentation for a Financial Application

Financial Application

The Datasets

In-Sample: 200 daily returns
Out-of-Sample: 20 daily returns
Rolling Windows: 11



Problem specification

Objective: $$\underset{\mathbf{w}}{minimize} \quad f(\mathbf{w}) = \sqrt{\frac{1}{T}\sum_{i=1}^{T}\left(R_t^P - R_t^I\right)^2} \qquad \mathbf{w} \in \Re^n$$

Constraints: *subject to*

$$(1)\ \sum_{i=1}^{n} w_i = 1$$

$$(2)\ 0 \le w_i \le 1$$

$$(3)\ 0.01 * \delta(w_i) \le w_i \le 0.1 * \delta(w_i),\ \delta(w_i) = \begin{cases} 1 & if\ w_i > 0 \\ 0 & otherwise \end{cases}$$

$$(4)\ 0 \le \sum_{i=1}^{n} \delta(w_i) \le K \quad K \in \{20, \ldots, 100\}$$

$$(5)\ \sum_{i: w_i > 0.05} w_i \le 0.4$$

$$(7) \sum_{i=1}^{n} | w_i - w_i | < 0.1$$

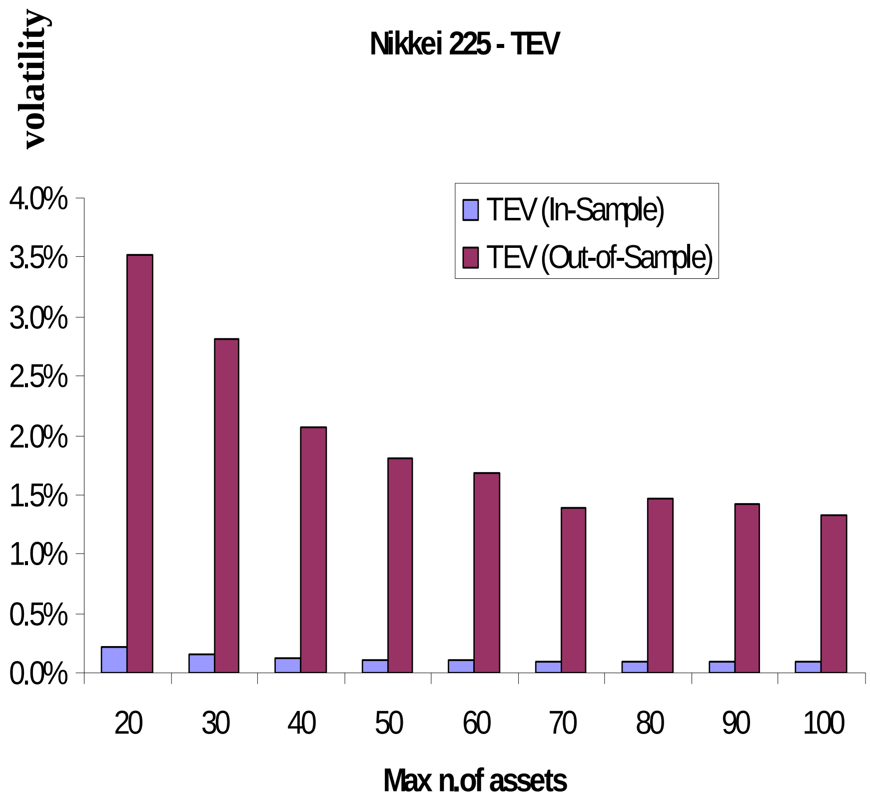
Financial Application

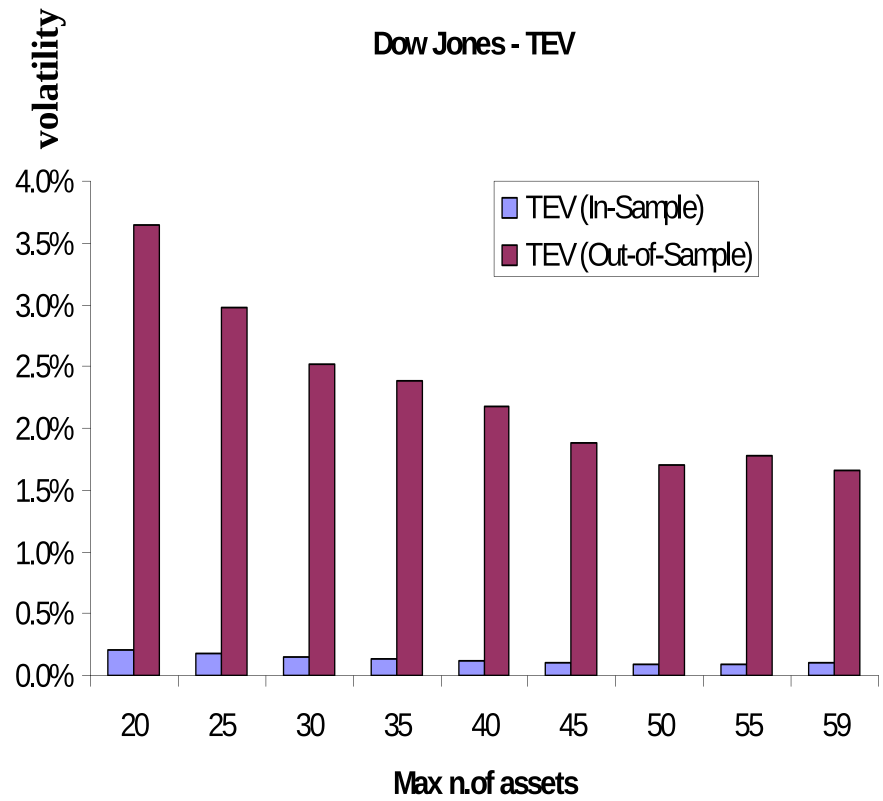
Statistics

- TEV: Annualized Average Tracking Error Volatility (In-Sample & Out-of-Sample)
- ER: Annualized Average Excess Return (In-Sample & Out-of-Sample)
- IR: Information Ratio (Out of Sample)
- Av.T.: Average Turnover ($\Sigma|\Delta w|/2$) (Out of Sample)
- Beta: (Out of Sample)
- Correlation: (Out of Sample)

Financial Application

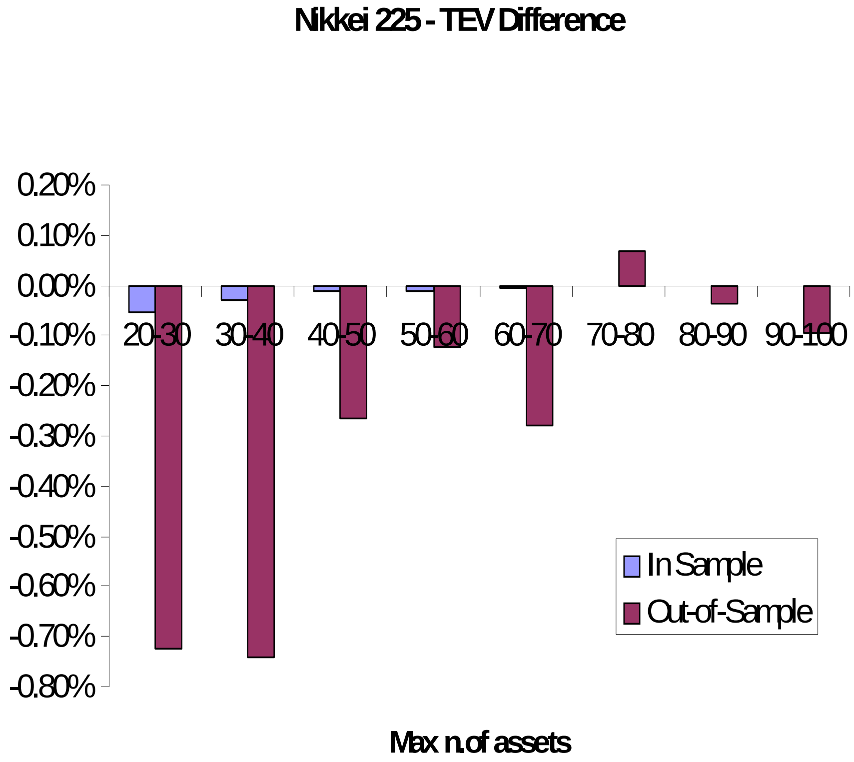
Tracking Error Volatility (TEV)

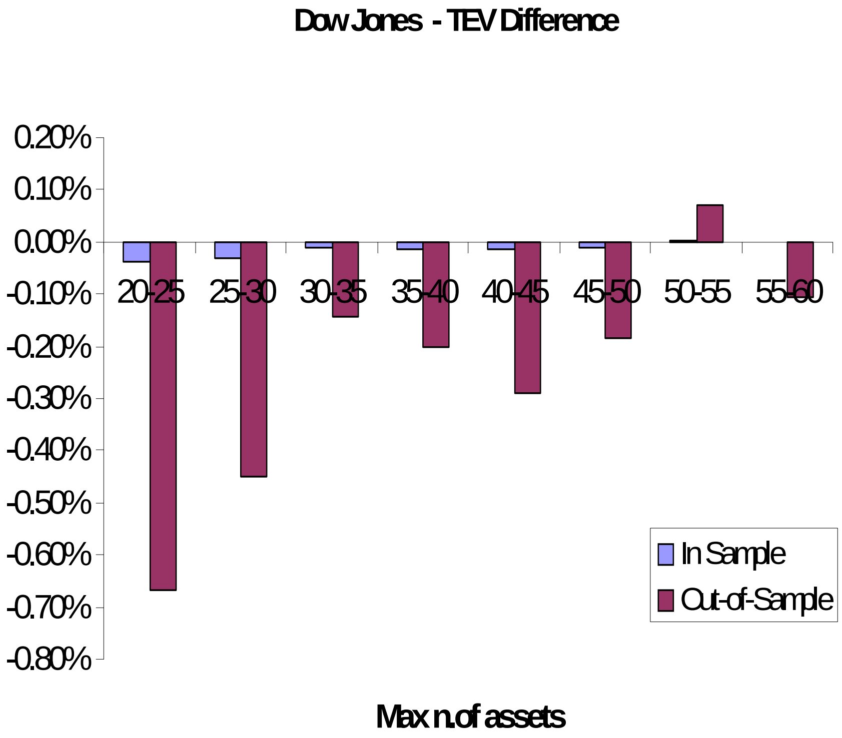




Financial Application

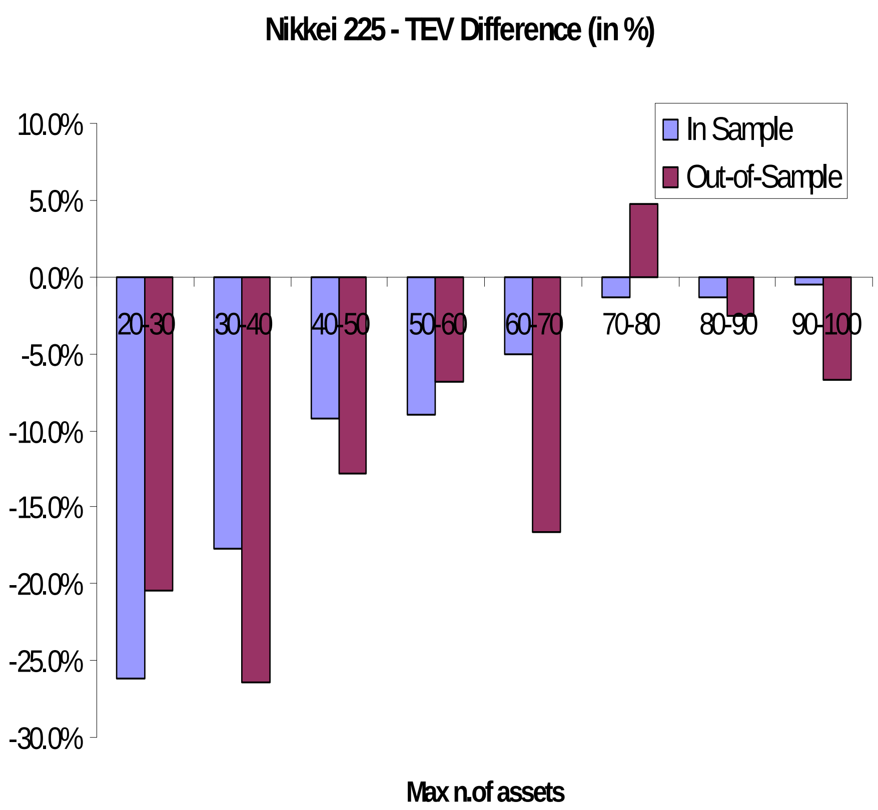
TEV Absolute Difference

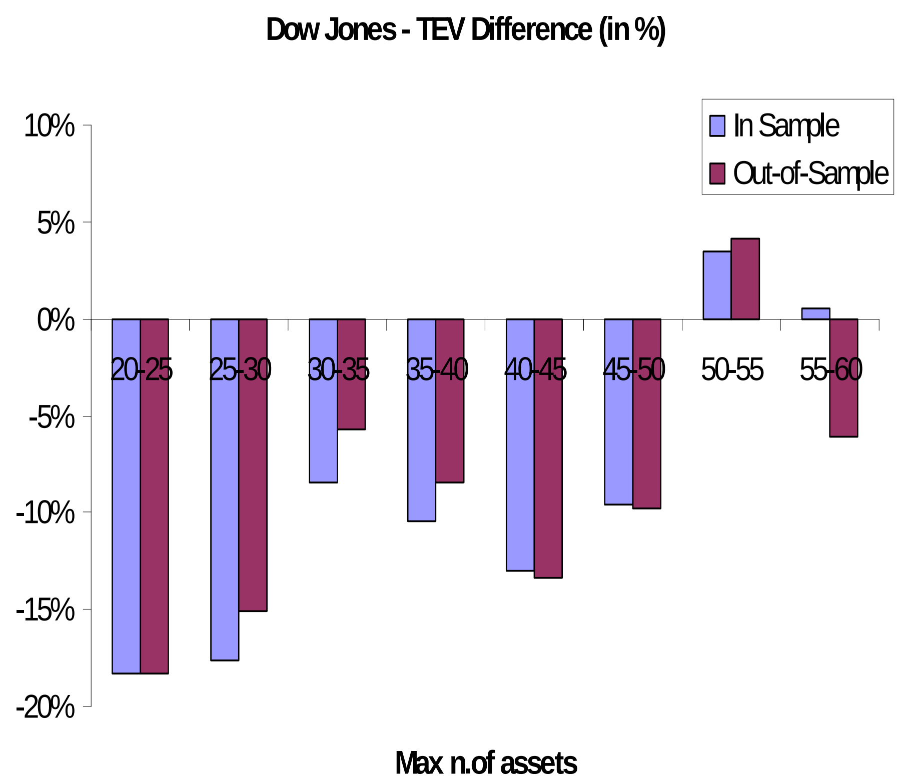




Financial Application

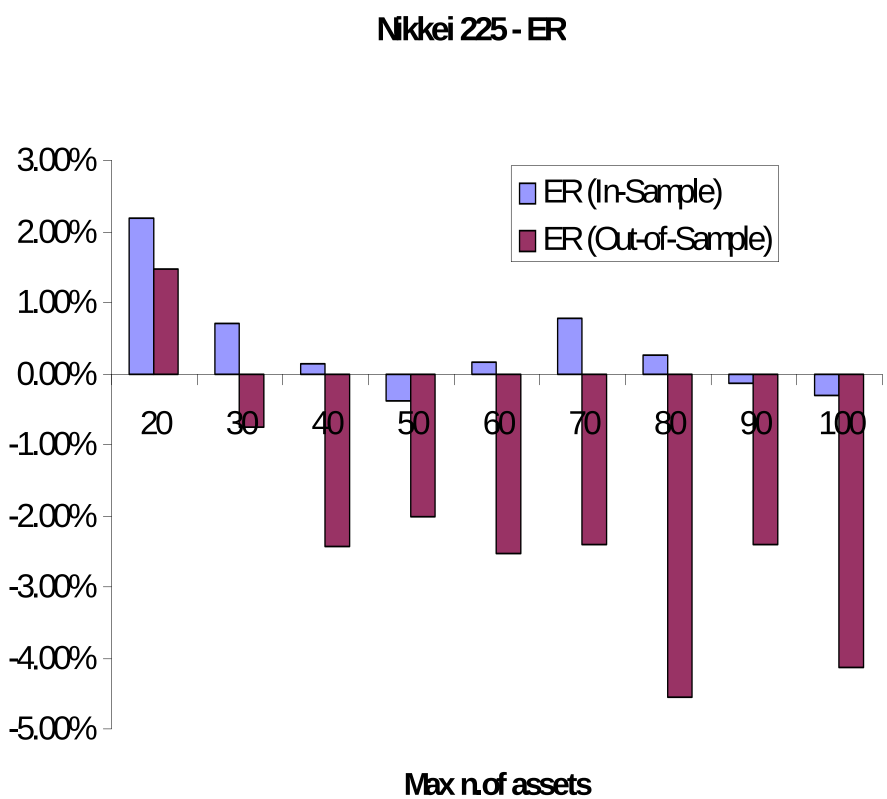
TEV Difference (in %)

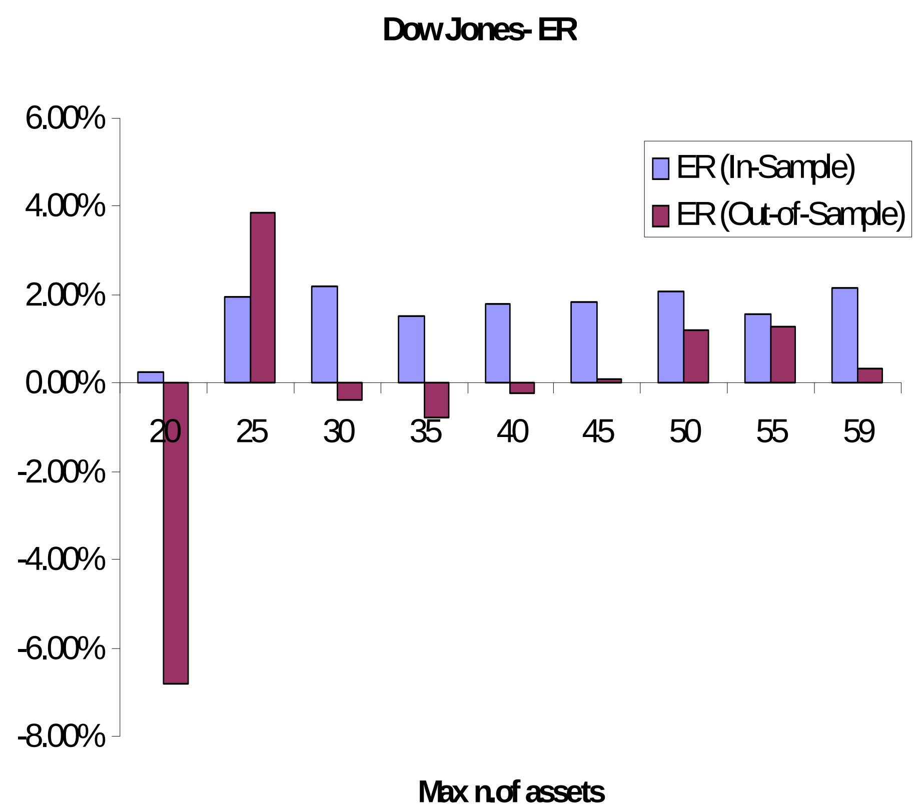




Financial Application

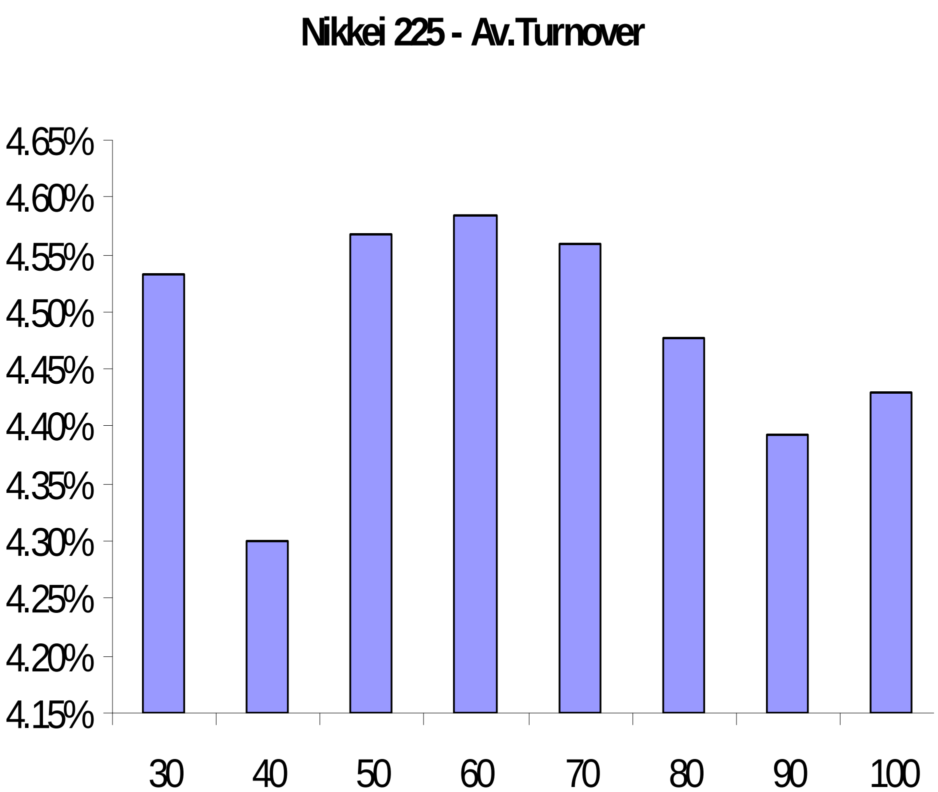
Excess Return

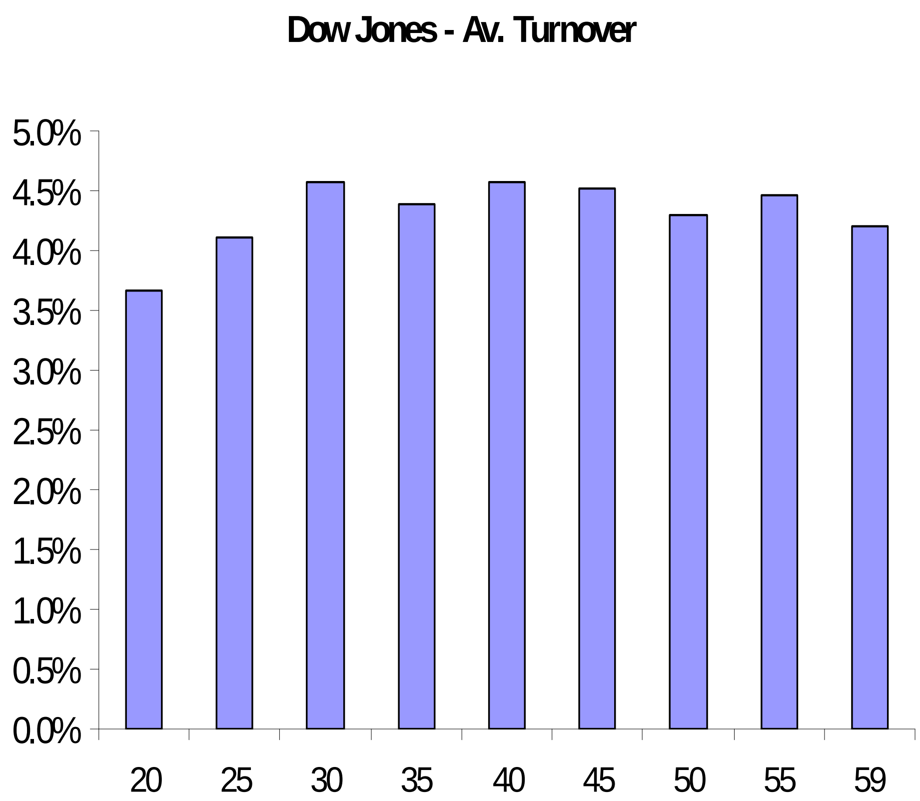




Financial Application

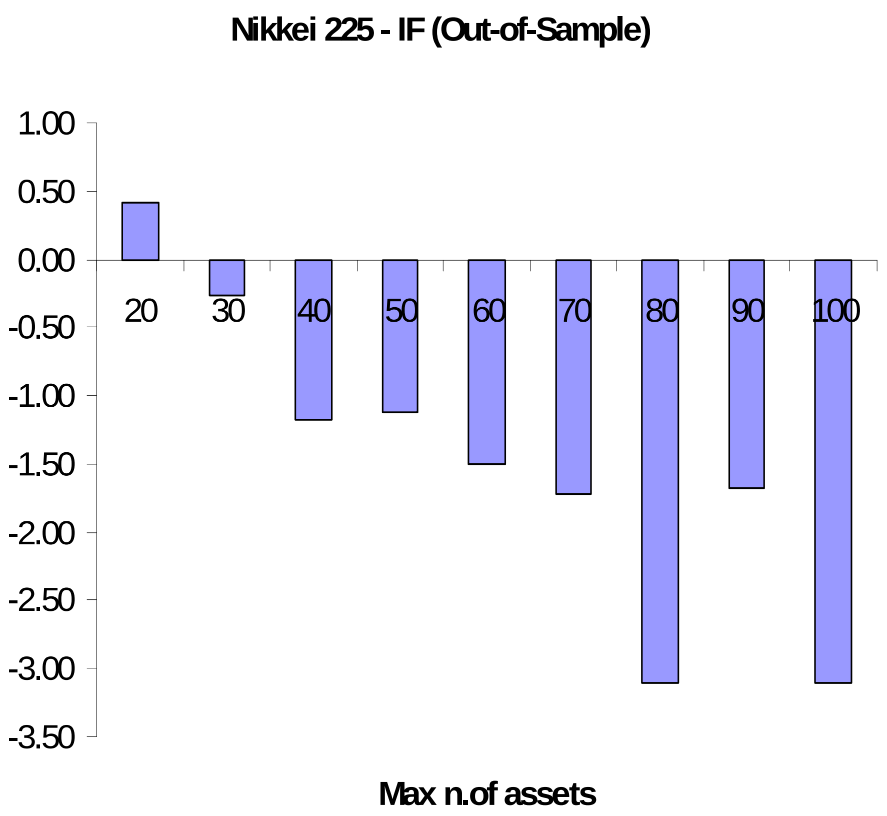
Av. Turnover

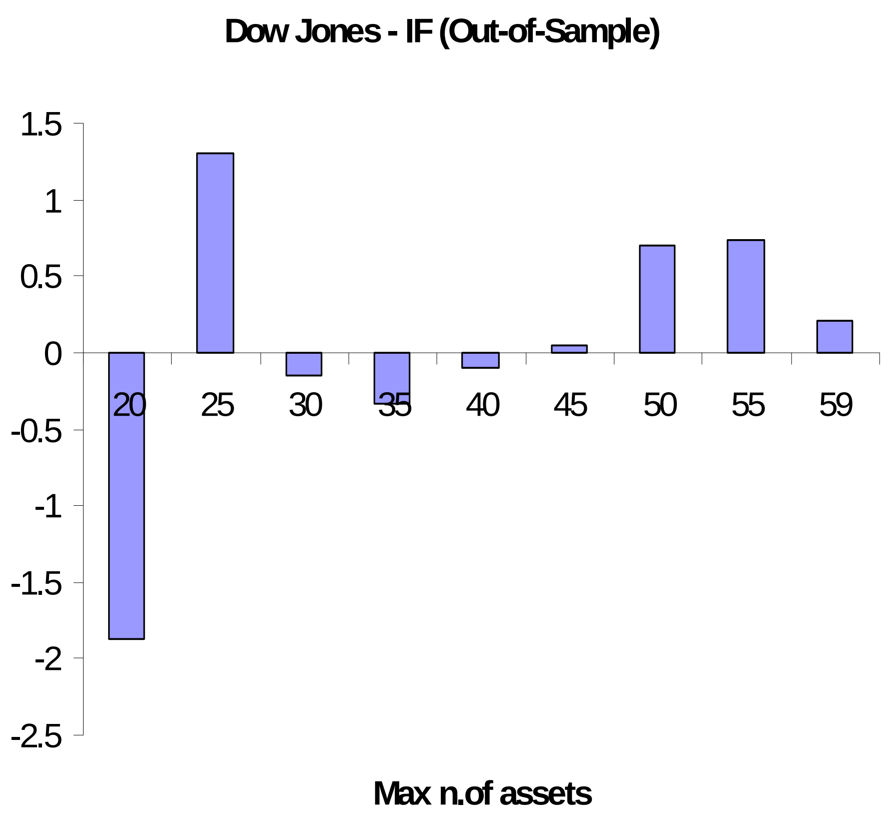




Financial Application

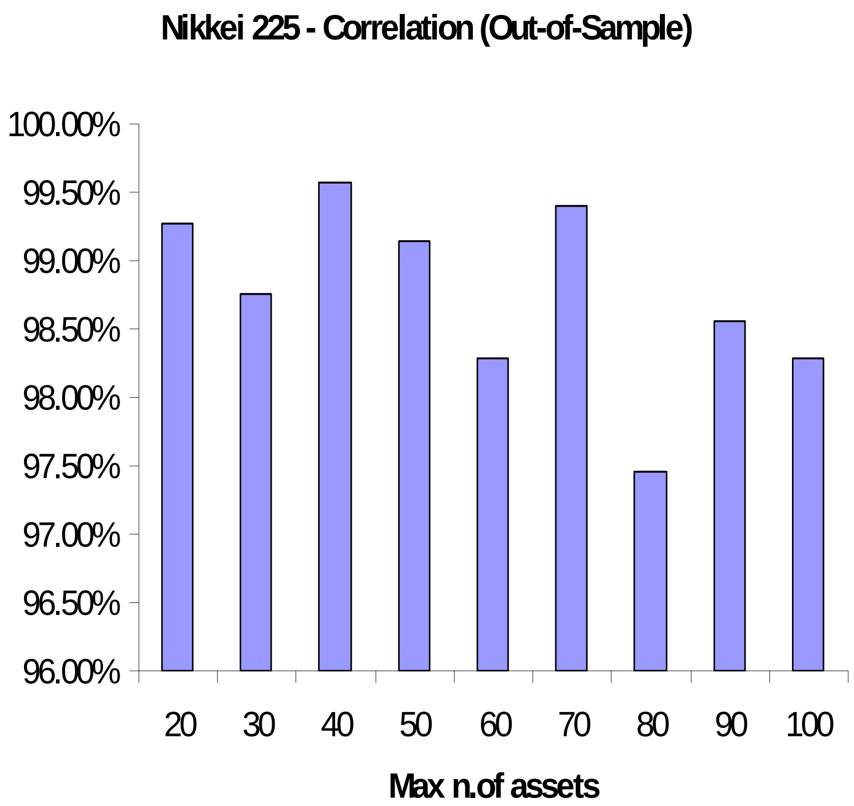
Information Ratio (ER/TEV)

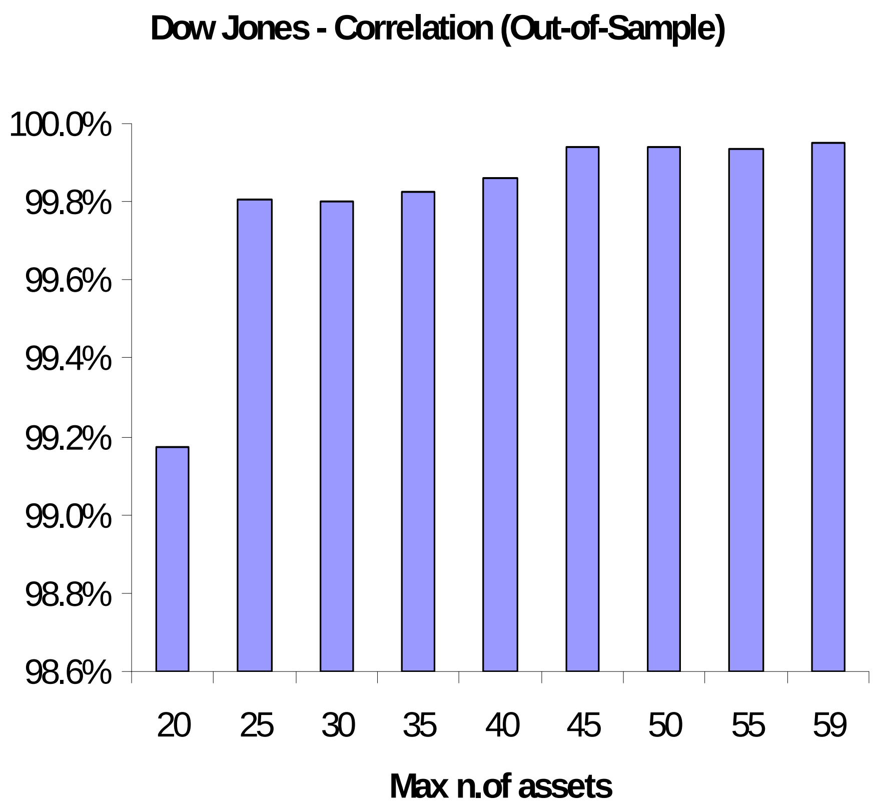




Financial Application

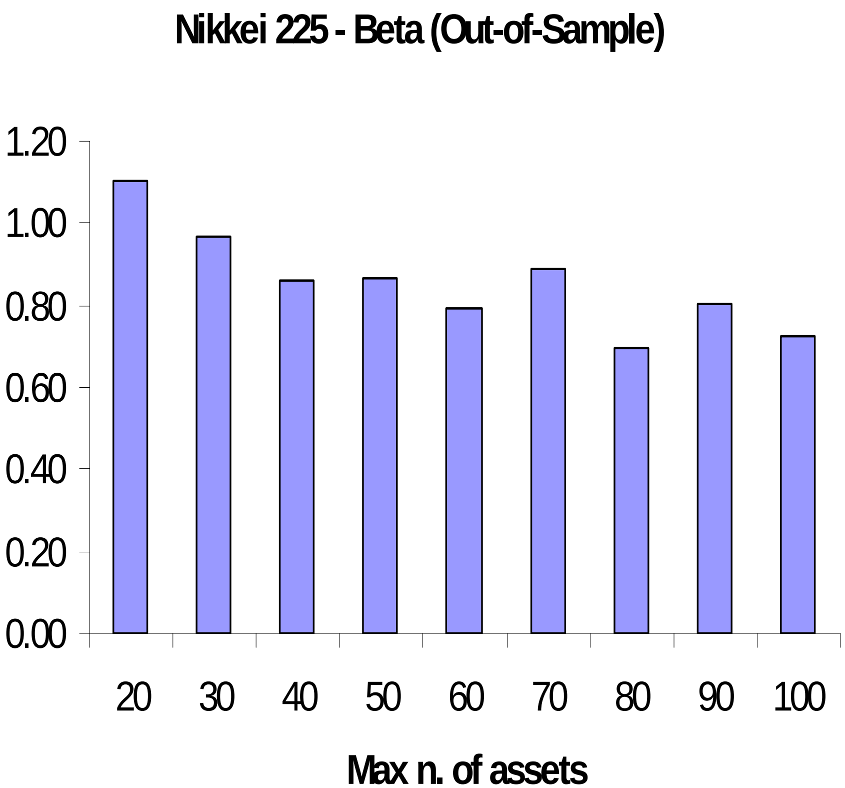
Correlation

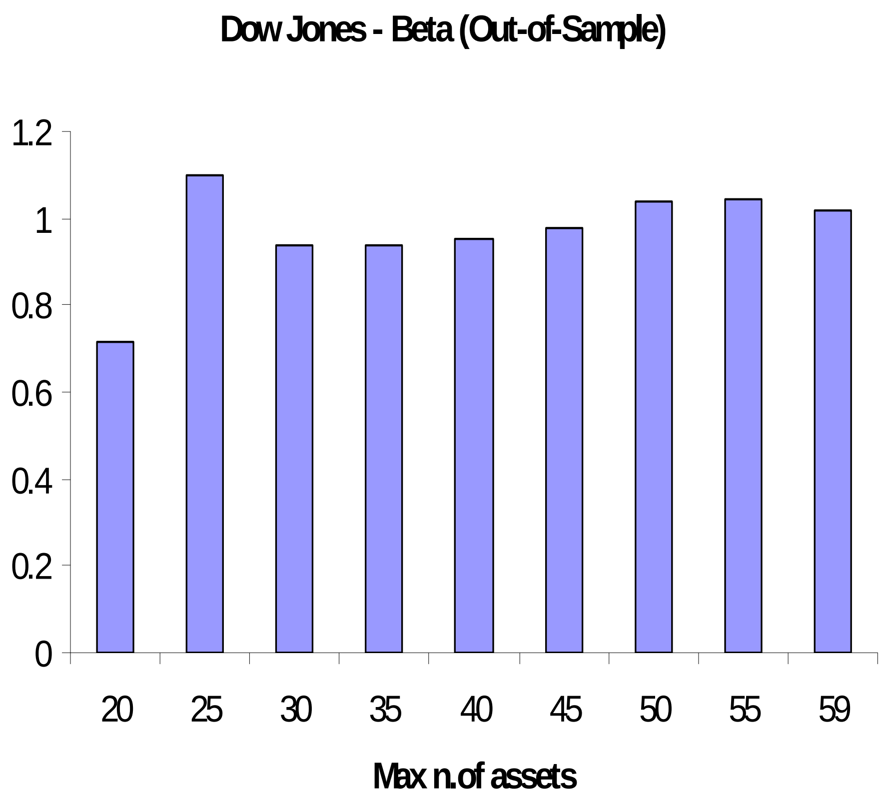




Financial Application

Beta





Conclusion

Summary

- DECS-IT can tackle the index tracking problem in a real-world context
- DECS-IT can easily consider other objective functions and constraints

Future Research

- Covariance estimation for index tracking
- Multi-objective optimization: min TEV and max ER
- Include transaction costs