Empirical Methods for the Analysis of Algorithms

Marco Chiarandini[1] Luís Paquete[2]

[1] Department of Mathematics and Computer Science
University of Southern Denmark, Odense, Denmark

[2] CISUC - Centre for Informatics and Systems of the University of Coimbra,
Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal

Semptember 7, 2007
Workshop on Engineering SLS Algorithms

Based on:
M. Chiarandini, L. Paquete, M. Preuss, E. Ridge (2007).
Experiments on metaheuristics: Methodological overview and open issues.
Tech. Rep. DMF-2007-03-003, The Danish Mathematical Society.

Outlook

The tutorial is about:

Applied Statistics
for Engineers and Scientists of SLS Algorithms

We aim at providing:

- basic notions of statistics
- a review of performance measures
- an overview of scenarios with proper tools for
  - exploratory data analysis
  - statistical inference

Outline

Outline

Outline

The Objects of Analysis

We consider search algorithms for solving (mainly) discrete optimization problems

For each general problem Π (e.g., TSP, GCP) we denote by C_Π a set (or class) of possible instances and by $\pi \in C_\Pi$ a single instance.

The object of analysis are SLS algorithms, *i.e.*, randomized search heuristics (with no "known" guarantee of optimality).

Definitions

More precisely:

- single-pass heuristics (denoted $\mathcal{A}^{\perp}$): have an embedded termination, for example, upon reaching a certain state

  (generalized optimization Las Vegas algorithms [Hoos and Stützle, 2004])

- asymptotic heuristics (denoted $\mathcal{A}^{\infty}$): do not have an embedded termination and they might improve their solution asymptotically

  (both probabilistically approximately complete and essentially incomplete [Hoos and Stützle, 2004])

Definitions

The most typical scenario considered in research on SLS algorithms:

Asymptotic heuristics with time limit decided *a priori*

The algorithm $\mathcal{A}^\infty$ is halted when time expires.

Deterministic case: $\mathcal{A}^\infty$ on π returns a solution of cost x.

The performance of $\mathcal{A}^\infty$ on π is a scalar $y = x$.

Randomized case: $\mathcal{A}^\infty$ on π returns a solution of cost X, where X is a random variable.

The performance of $\mathcal{A}^\infty$ on π is the univariate $Y = X$.

[This is not the only relevant scenario: to be refined later]

Generalization

On a specific instance, the random variable Y that defines the performance measure of an algorithm is described by its probability distribution/density function

$$p(y|\pi)$$

It is often more interesting to generalize the performance on a class of instances C_Π, that is,

$$p(y, C_\Pi) = \sum_{\pi \in \Pi} p(y|\pi)p(\pi)$$

Outline

Theory *vs* Practice

Task: explain the performance of algorithms

Theoretical Analysis:

- worst case analysis: considers all possible problem instances of a problem
- average case analysis: assumes knowledge on the distribution of problem instances.

But:

- results may have low practical relevance
- problems and algorithms are complex

Experimental Analysis:

- it is (often) easy and fast to collect data
- results are fast and have practical relevance

Experimental Algorithmics

Experimental Algorithmics: *"is concerned with the design, implementation, tuning, debugging and performance analysis of computer programs for solving algorithmic problems"*.

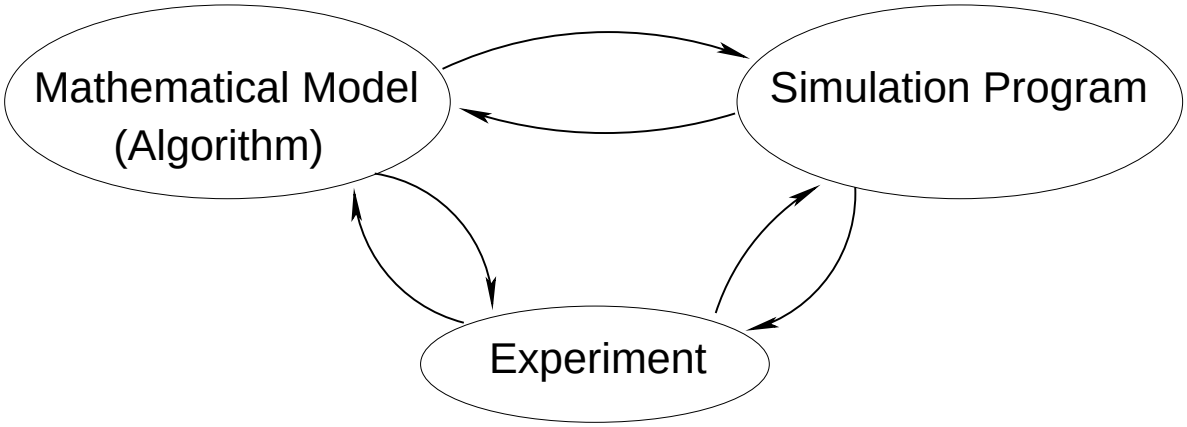
[Demetrescu and Italiano, 2000]

Looks at algorithms as a problem of the natural sciences instead of "only" as a mathematical problem.

Goals

- Defining standard methodologies
- Identifying and collecting problem instances from the real-world and instance generators
- Comparing relative performance of algorithms so as to identify the best ones for a given application
- Identifying algorithm separators, *i.e.*, families of problem instances for which the performance differ
- Providing new insights in algorithm design

Experimental Algorithmics



[McGeoch, 1996]

Algorithmic models of programs can vary according to their level of instantiation:

- minimally instantiated (algorithmic framework), e.g., simulated annealing
- mildly instantiated: includes implementation strategies (data structures)
- highly instantiated: includes details specific to a particular programming language or computer architecture

Sampling

In experiments,

- we sample the population of instances and
- we sample the performance of the algorithm on each sampled instance

If on an instance π we run the algorithm r times then we have r replicates of the performance measure Y, denoted $Y_1, \ldots, Y_r$, which are independent and identically distributed (i.i.d.), i.e.

$$p(y_1, \ldots, y_r|\pi) = \prod_{j=1}^{r} p(y_j|\pi)$$

$$p(y_1, \ldots, y_r) = \sum_{\pi \in C_\Pi} p(y_1, \ldots, y_r|\pi)p(\pi).$$

Test Instance Selection

In real-life applications a simulation of $p(\pi)$ can be obtained by historical data.

In research studies instances may be:

- real world instances
- random variants of real world-instances
- online libraries
- randomly generated instances

They may be grouped in classes according to some features whose impact may be worth studying:

- type (for features that might impact performance)
- size (for scaling studies)
- hardness (focus on hard instances)
- application (e.g., CSP encodings of scheduling problems), ...

Within the class, instances are drawn with uniform probability $p(\pi) = c$

Statistical Methods

The analysis of performance is based on finite-sized sampled data. Statistics provides the methods and the mathematical basis to

- describe, summarizing, the data (descriptive statistics)
- make inference on those data (inferential statistics)

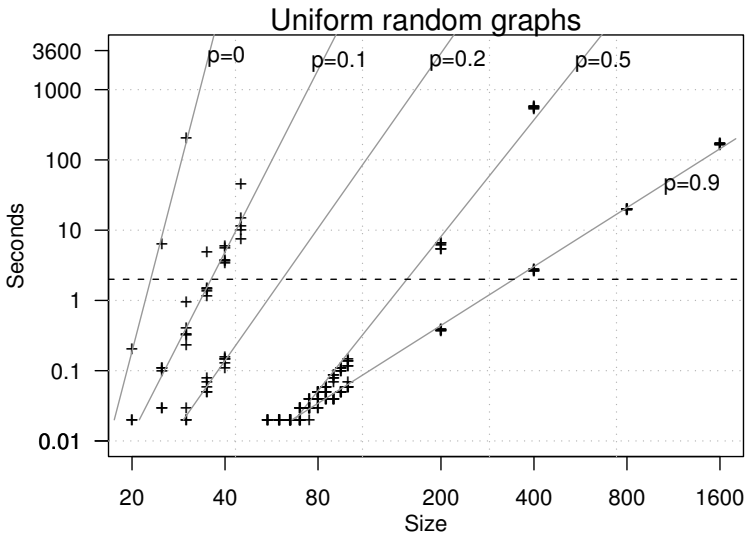
In research, statistics helps to

- guarantee replicability
- make results reliable
- help to extract relevant results from large amount of data

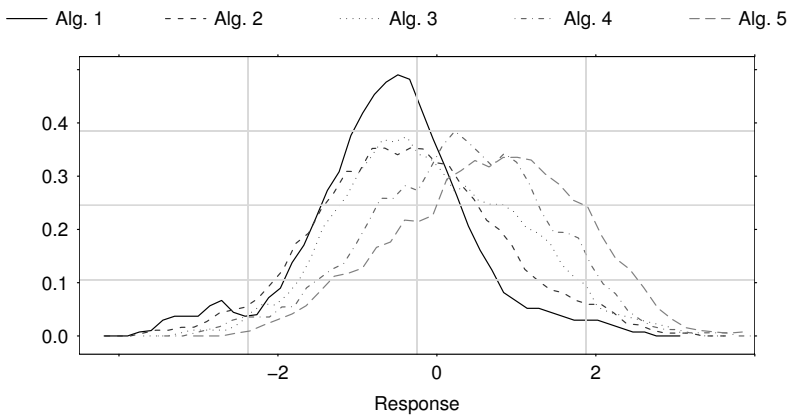
In the practical context of heuristic design and implementation (i.e., engineering), statistics helps to take sound decisions with the least amount of experimentation

Objectives of the Experiments

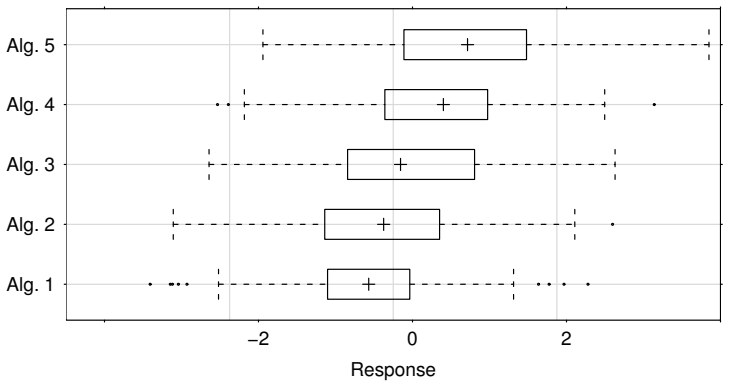
- **Characterization**:
  Interpolation: fitting models to data
  Extrapolation: building models of data, explaining phenomena
  - Standard statistical methods: *linear and non linear regression*
    model fitting



Objectives of the Experiments



- **Comparison**: bigger/smaller, same/different, Algorithm Configuration, Component-Based Analysis
  - Standard statistical methods: *experimental designs, test hypothesis and estimation*



Outline

Measures and Transformations

On a single instance

Computational effort indicators

- CPU time (real time as measured by OS functions)
- number of elementary operations/algorithmic iterations (e.g., search steps, objective function evaluations, number of visited nodes in the search tree, consistency checks, etc.)

Solution quality indicators

- value returned by the cost function (or error from optimum/reference value)

Measures and Transformations

On a class of instances

Computational effort indicators

- no transformation if the interest is in studying scaling
- standardization if a fixed time limit is used
- otherwise, better to group homogeneously the instances

Solution quality indicators

Different instances implies different scales $\Rightarrow$ need for an invariant measure

But also, see [McGeoch, 1996].

Measures and Transformations

On a class of instances

Solution quality indicators

- Distance or error from a reference value (assume minimization case):

$$e_1(x,\pi) = \frac{x(\pi) - \bar{x}(\pi)}{\sqrt{\widehat{\sigma(\pi)}}} \qquad \text{standard score}$$

$$e_2(x,\pi) = \frac{x(\pi) - x^{opt}(\pi)}{x^{opt}(\pi)} \qquad \text{relative error}$$

$$e_3(x,\pi) = \frac{x(\pi) - x^{opt}(\pi)}{x^{worst}(\pi) - x^{opt}(\pi)} \qquad \text{invariant [Zemel, 1981]}$$

  - optimal value computed exactly or known by instance construction
  - surrogate value such bounds or best known values
- Rank (no need for standardization but loss of information)

Outline

Outline

Scenarios (Refinement)

- Single-pass heuristics
- Asymptotic heuristics:
  Three approaches:
  1. Univariate
     - 1.a Time as an external parameter decided *a priori*
     - 1.b Solution quality as an external parameter decided *a priori*
  2. Cost dependent on running time:
  3. Cost and running time as two minimizing objectives

Definitions

Single-pass heuristics

Deterministic case: $\mathcal{A}^{\perp}$ on π returns a solution of cost x with computational effort t (*e.g.*, running time).

The performance of $\mathcal{A}^{\perp}$ on π is the vector $\vec{y} = (x, t)$.

Randomized case: $\mathcal{A}^{\perp}$ on π returns a solution of cost X with computational effort T, where X and T are random variables.

The performance of $\mathcal{A}^{\perp}$ on π is the bivariate $\vec{Y} = (X, T)$.

Single-pass Heuristics

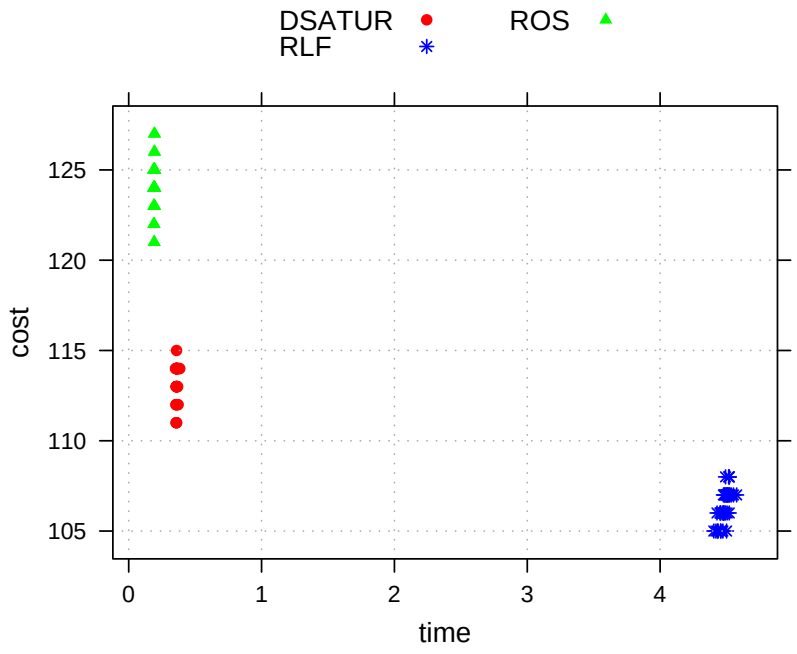
Bivariate analysis: Example

Scenario:

- ▷ 3 heuristics $\mathcal{A}_1^{\perp}$, $\mathcal{A}_2^{\perp}$, $\mathcal{A}_3^{\perp}$ on class C_{Π}.
- ▷ homogeneous instances or need for data transformation.
- ▷ 1 or r runs per instance
- ▶ Interest: inspecting solution cost and running time to observe and compare the level of approximation and the speed.

Tools:

- ▶ Scatter plots of solution-cost and run-time



Definitions

Asymptotic heuristics

There are three approaches:

1.a. Time as an external parameter decided *a priori*.
The algorithm is halted when time expires.

Deterministic case: $\mathcal{A}^\infty$ on π returns a solution of cost x.

The performance of $\mathcal{A}^\infty$ on π is the scalar $y = x$.

Randomized case: $\mathcal{A}^\infty$ on π returns a solution of cost X, where X is a random variable.

The performance of $\mathcal{A}^\infty$ on π is the univariate $Y = X$.

Asymptotic Heuristics

Approach 1.a, Univariate analysis

Scenario:

- ▷ 3 heuristics $\mathcal{A}_1^\infty$, $\mathcal{A}_2^\infty$, $\mathcal{A}_3^\infty$ on class C_Π.
  (Or 3 heuristics $\mathcal{A}_1^\infty$, $\mathcal{A}_2^\infty$, $\mathcal{A}_3^\infty$ on class C_Π without interest in computation time because negligible or comparable)
- ▷ homogeneous instances (no data transformation) or heterogeneous (data transformation)
- ▷ 1 or $\mathfrak{r}$ runs per instance
- ▷ a priori time limit imposed
- ▶ Interest: inspecting solution cost

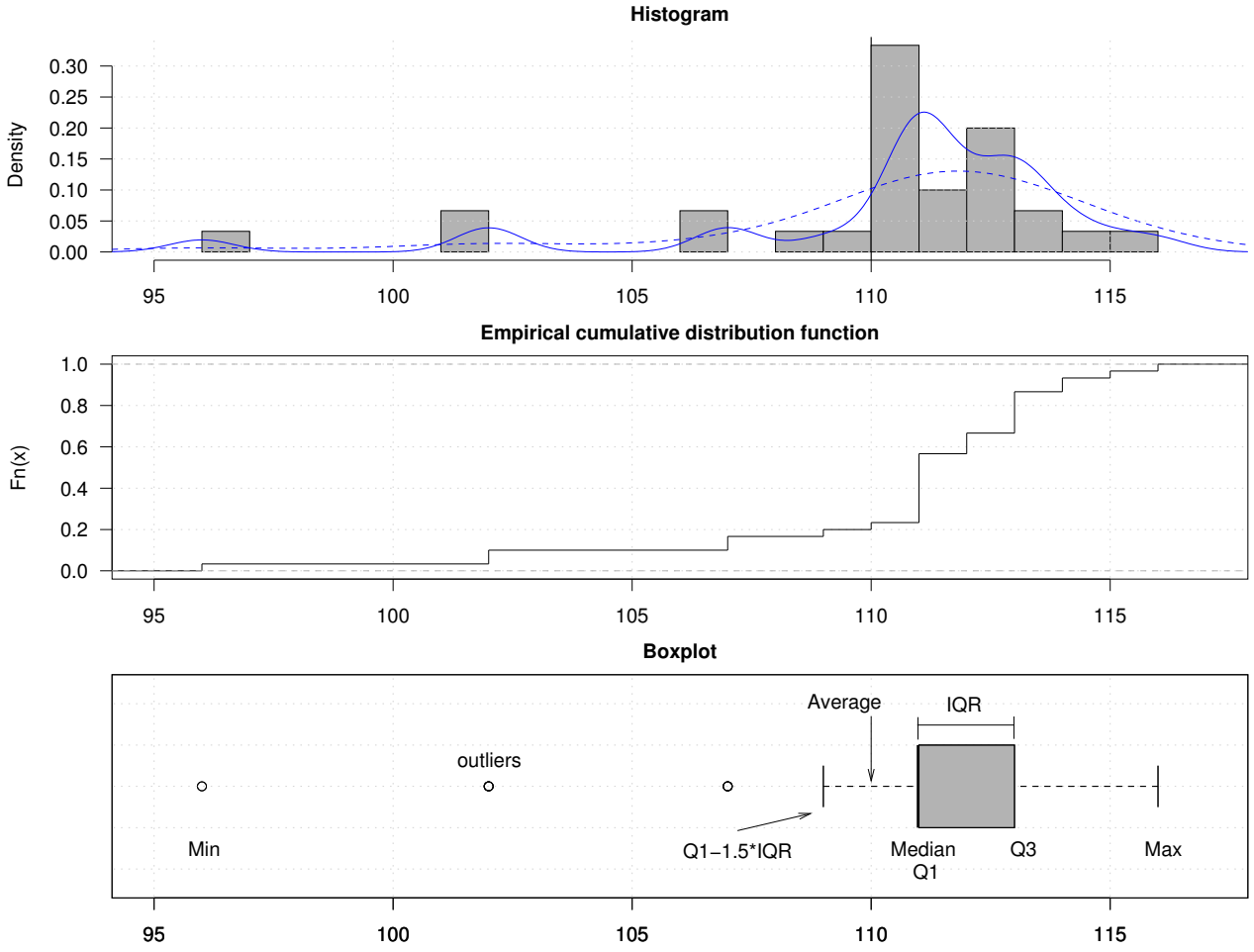
Tools:

- Histograms (summary measures: mean or median or mode?)
- Boxplots
- Empirical cumulative distribution functions (ECDFs)

Asymptotic Heuristics

Approach 1.a, Univariate analysis: Example

Data representation

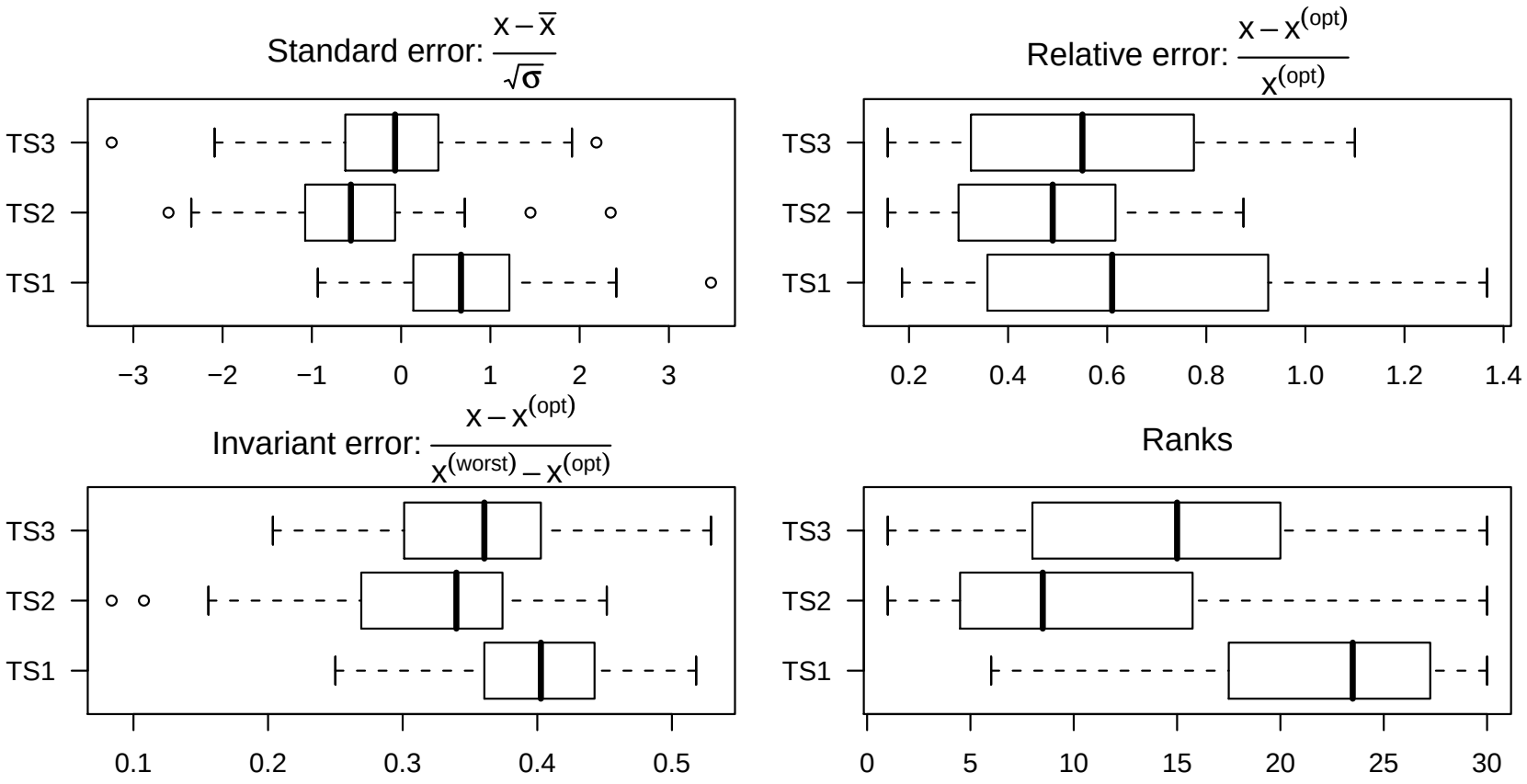


Measures of central tendency (mean, median, mode)
and dispersion (variance, standard deviation, inter-quartile)

Asymptotic Heuristics

Approach 1.a, Univariate analysis: Example

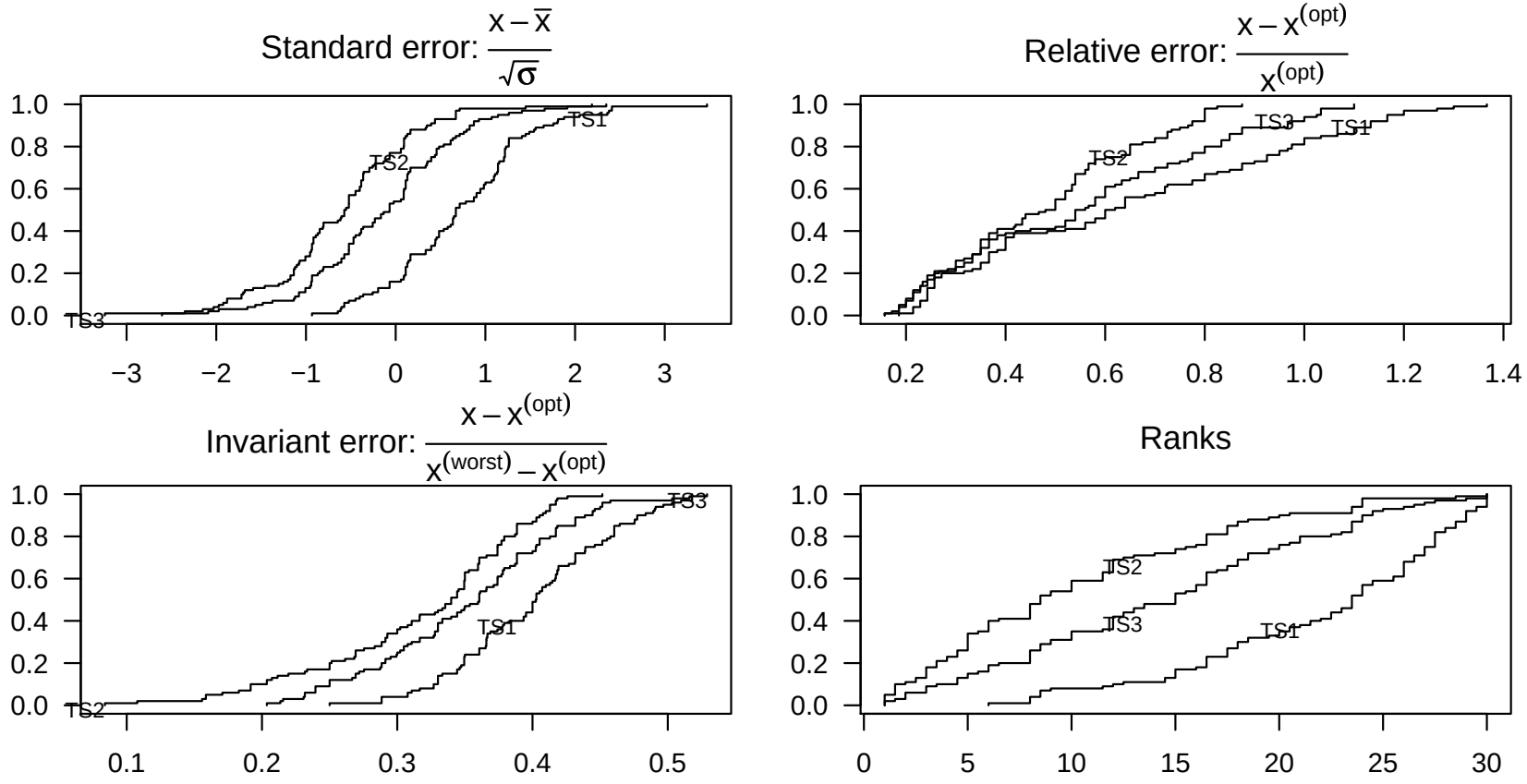
On a class of instances



Asymptotic Heuristics

Approach 1.a, Univariate analysis: Example

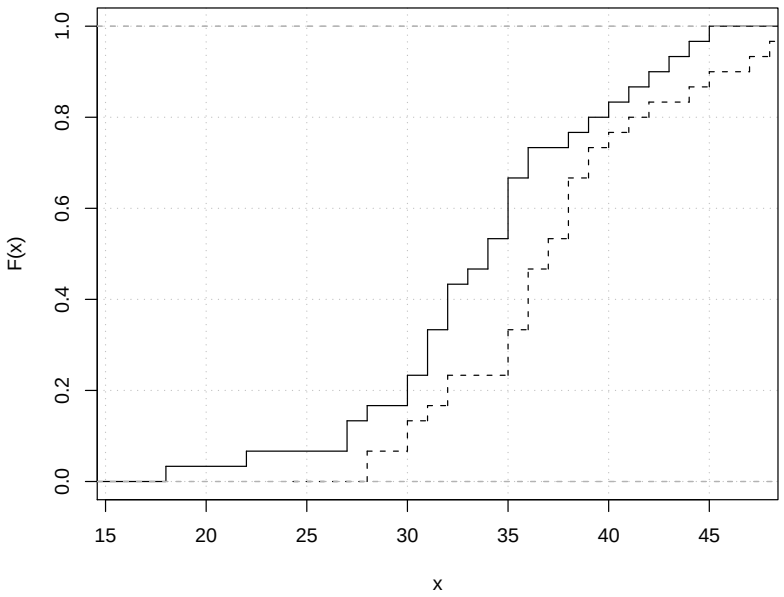
On a class of instances

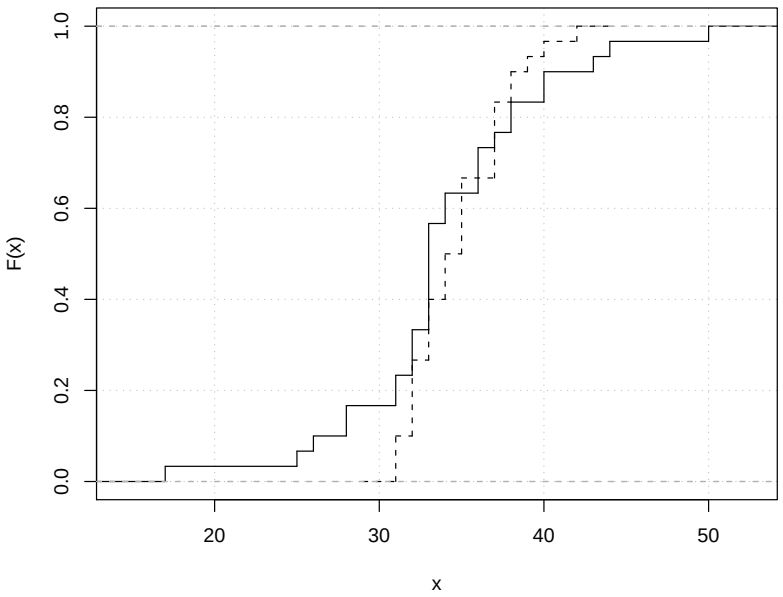


Stochastic Dominance

Definition: Algorithm $\mathcal{A}_1$ probabilistically dominates algorithm $\mathcal{A}_2$ on a problem instance, iff its CDF is always "below" that of $\mathcal{A}_2$, *i.e.*:

$$F_1(x) \leq F_2(x), \qquad \forall x \in X$$





Definitions

Asymptotic heuristics

There are three approaches:

1.b. Solution quality as an external parameter decided *a priori*. The algorithm is halted when quality is reached.

Deterministic case: $\mathcal{A}^\infty$ on π finds a solution in running time t.

The performance of $\mathcal{A}^\infty$ on π is the scalar $y = t$.

Randomized case: $\mathcal{A}^\infty$ on π finds a solution in running time T, where T is a random variable.

The performance of $\mathcal{A}^\infty$ on π is the univariate $Y = T$.

Dealing with Censored Data

Asymptotic heuristics, Approach 1.b

- ▷ Heuristics $\mathcal{A}^{\perp}$ stopped before completion or $\mathcal{A}^{\infty}$ truncated (always the case)
- ▶ Interest: determining whether a prefixed goal (optimal/feasible) has been reached

The computational effort to attain the goal can be specified by a cumulative distribution function $F(t) = P(T < t)$ with T in $[0, \infty)$.

If in a run i we stop the algorithm at time L_i then we have a Type I right censoring, that is, we know either

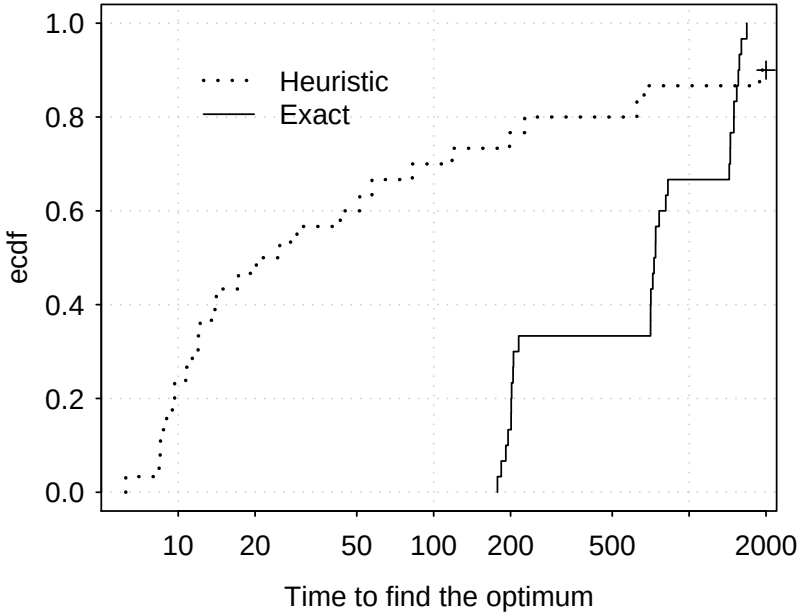
- T_i if $T_i \leq L_i$
- or $T_i \geq L_i$.

Hence, for each run i we need to record $\min(T_i, L_i)$ and the indicator variable for observed optimal/feasible solution attainment, $\delta_i = I(T_i \leq L_i)$.

Dealing with Censored Data

Asymptotic heuristics, Approach 1.b: Example

- An exact *vs* an heuristic algorithm for the *2-Edge-connectivity augmentation problem*.
- Interest: time to find the optimum on different instances.



Uncensored:

$$F(t) = \frac{\#\ \text{runs} < t}{n}$$

Censored:

$$F(t) = 1 - \prod_{j:t_j<t} \frac{n_j - d_j}{n_j} \qquad t \in [0, L]$$

(Kaplan-Meier estimation model)

Definitions

Asymptotic heuristics

There are three approaches:

2. Cost dependent on running time:

Deterministic case: $\mathcal{A}^\infty$ on π returns a current best solution x at each observation in $t_1, \ldots, t_k$.

The performance of $\mathcal{A}^\infty$ on π is the profile indicated by the vector $\vec{y} = \{x(t_1), \ldots, x(t_k)\}$.

Randomized case: $\mathcal{A}^\infty$ on π produces a monotone stochastic process in solution cost $X(\tau)$ with any element dependent on the predecessors.

The performance of $\mathcal{A}^\infty$ on π is the multivariate $\vec{Y} = (X(t_1), X(t_2), \ldots, X(t_k))$.

Asymptotic Heuristics

Approach 2, Multivariate analysis

Scenario:

- ▷ 3 heuristics $\mathcal{A}_1^\infty$, $\mathcal{A}'^\infty_2$, $\mathcal{A}_3^\infty$ on instance π.
- ▷ single instance hence no data transformation.
- ▷ r runs
- ▶ Interest: inspecting solution cost over running time to determine whether the comparison varies over time intervals

Tools:

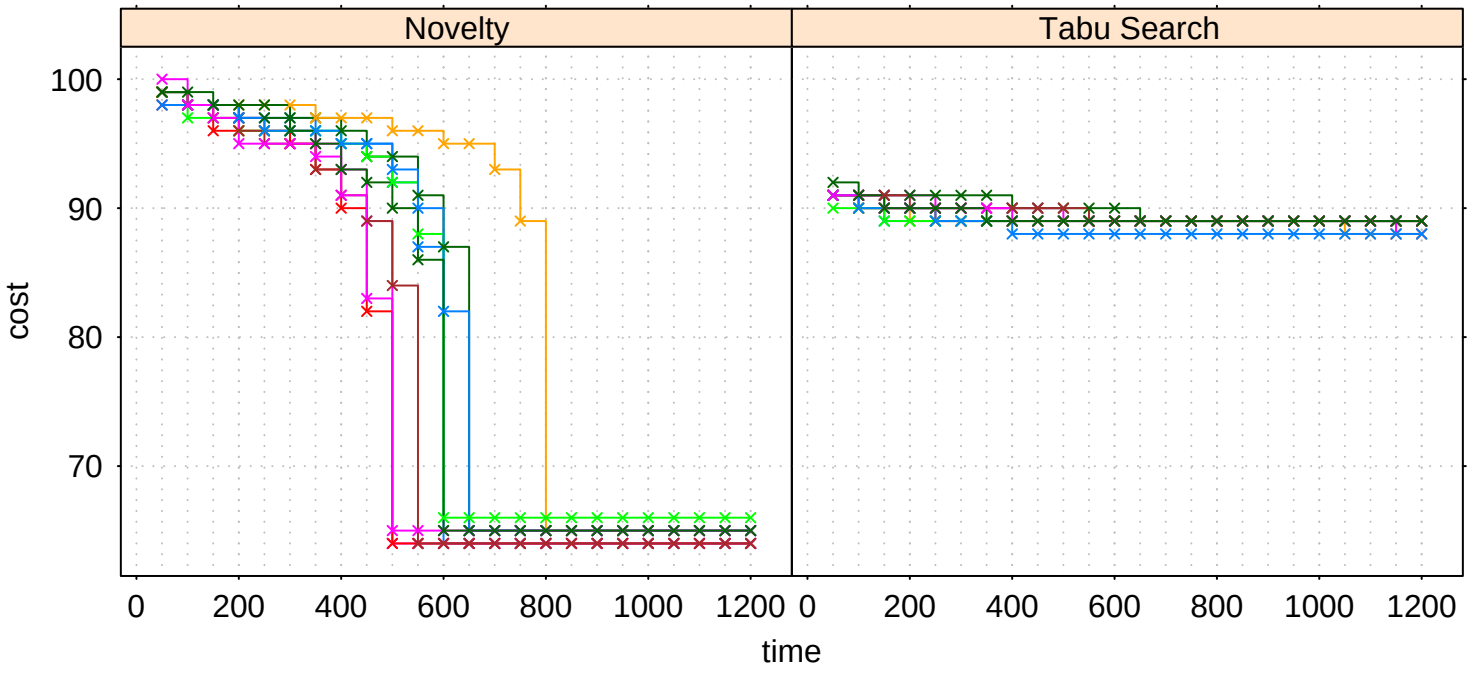
- ▶ Quality profiles

Asymptotic Heuristics

Approach 2, Multivariate analysis: Example

The performance is described by multivariate random variables of the kind $\vec{Y} = \{Y(t_1), Y(t_2), \ldots, Y(l_k)\}$.

Sampled data are of the form $\vec{Y_i} = \{Y_i(t_1), Y_i(t_2), \ldots, Y_i(t_k)\}$, $i = 1, \ldots, 10$ (10 runs per algorithm on one instance)

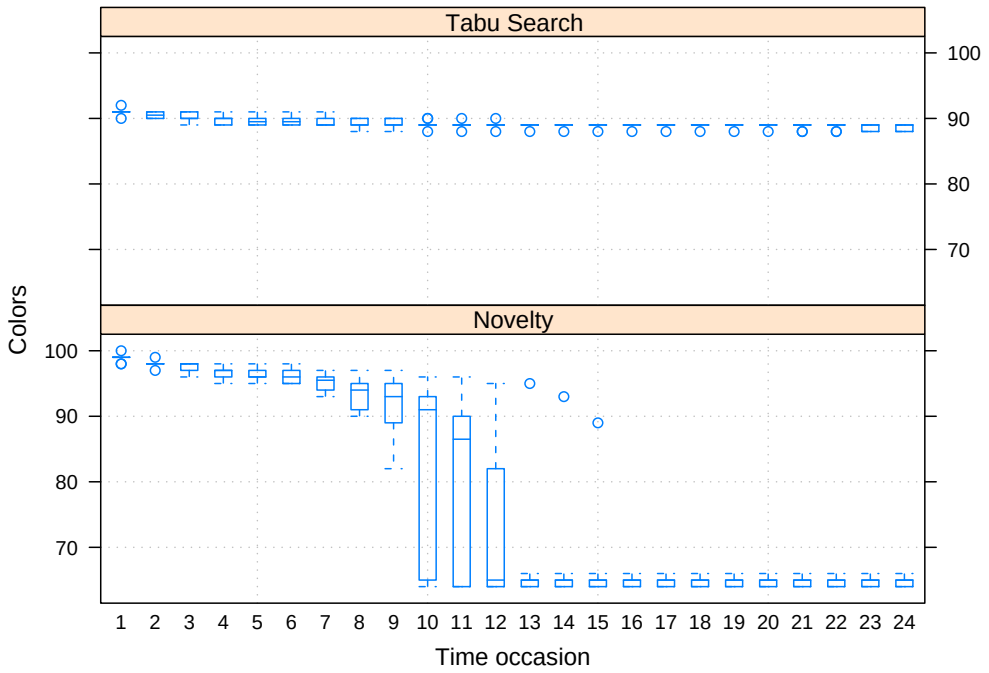


Asymptotic Heuristics

Approach 2, Multivariate analysis: Example

The performance is described by multivariate random variables of the kind $\vec{Y} = \{Y(t_1), Y(t_2), \ldots, Y(l_k)\}$.

Sampled data are of the form $\vec{Y_i} = \{Y_i(t_1), Y_i(t_2), \ldots, Y_i(t_k)\}$, $i = 1, \ldots, 10$ (10 runs per algorithm on one instance)

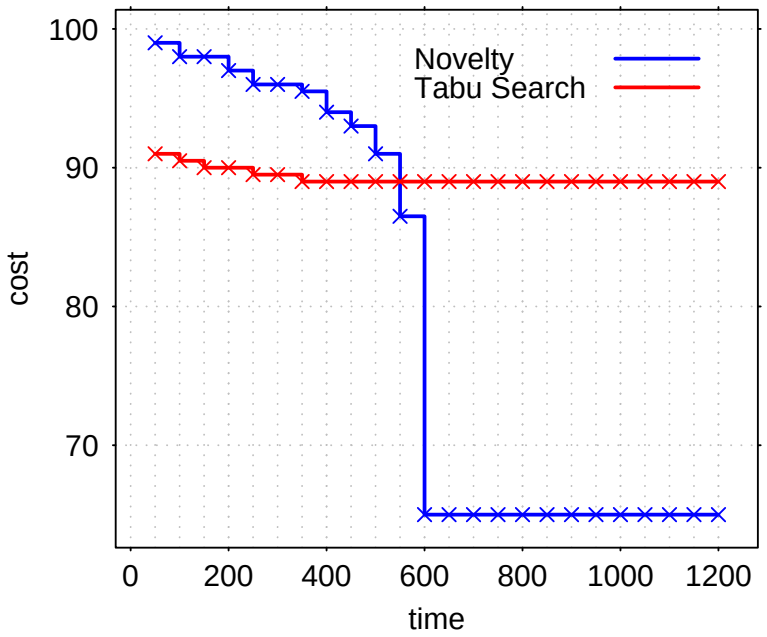


Asymptotic Heuristics

Approach 2, Multivariate analysis: Example

The performance is described by multivariate random variables of the kind $\vec{Y} = \{Y(t_1), Y(t_2), \ldots, Y(l_k)\}$.

Sampled data are of the form $\vec{Y_i} = \{Y_i(t_1), Y_i(t_2), \ldots, Y_i(t_k)\}$, $i = 1, \ldots, 10$ (10 runs per algorithm on one instance)



The median behavior of the two algorithms

Definitions

Asymptotic heuristics

There are three approaches:

3. Cost and running time as two minimizing objectives [da Fonseca et al., 2001]:

Deterministic case: $\mathcal{A}^\infty$ on π finds improvements in solution cost which are represented by a set of *non-dominated* points $\mathcal{Y} = \{\vec{y}_j \in \mathbf{R}^2, j = 1, \ldots, m\}$ of solution cost and running time

Randomized case: $\mathcal{A}^\infty$ on π finds improvements in solution cost which are represented by a random variable, *i.e.*, a random set of *non-dominated* points $\mathcal{Y} = \{\vec{Y}_j \in \mathbf{R}^2, j = 1, \ldots, m\}$.

The performance of $\mathcal{A}^\infty$ on π is measured by the random set $\mathcal{Y}$

Asymptotic Heuristics

Approach 3, Bi-objective analysis

Needed some definitions on dominance relations

In a run on π, $\mathcal{A}^\infty$ passes through states, *i.e.*, a set of points $y = \{\vec{y}_j \in \mathbf{R}^2, j = 1, \ldots, m\}$, each one being a vector of solution cost and running time.

In Pareto sense, for points in $\mathbf{R}^2$

$\vec{y}^1 \preceq \vec{y}^2$	weakly dominates	$y_i^1 \leq y_i^2$ for all $i = 1, 2$
$\vec{y}^1 \parallel \vec{y}^2$	incomparable	neither $\vec{y}^1 \preceq \vec{y}^2$ nor $\vec{y}^2 \preceq \vec{y}^1$

Hence we can resume the run of the algorithm by a set of mutually incomparable points (*i.e.*, weakly non-dominated points)

Extended to sets of points:

$A \preceq_s B$	weakly dominates	for all $\vec{b} \in B$ exists $\vec{a} \in A$ such that $\vec{a} \preceq \vec{b}$
$A \parallel B$	incomparable	neither $A \preceq_S B$ nor $B \preceq_S A$

Performance Measures

Multi-Objective Case

How to compare the approximation sets from multiple runs of two or more multiobjective algorithms?

For two approximation sets $A, B \in \Omega$ (Ω set of all approximation sets) dominance relations may indicate:

A is better then B	A and B are incomparable
B is better then A	A and B are indifferent

Other Pareto compliant quantitative indicators have been introduced

- Unary indicators ($I : \Omega \to \mathbf{R}$) (need a reference point) and
  - Hypervolume
  - Epsilon indicator
  - R indicator
- Binary indicators ($I : \Omega \times \Omega \to \mathbf{R}$):
  - Epsilon indicator
- Ranking
- Attainment function

Note that with unary indicators, if $I(A) < I(B)$, we can only say that A is not worse than B (they may be incomparable) [Zitzler et al., 2003].

Asymptotic Heuristics

Approach 3, Bi-objective analysis

Let $\mathcal{Y} = \{\vec{Y}_j \in \mathbf{R}^2, j = 1, \ldots, m\}$ be a random set of m mutually independent points of solution-cost and run-time

The attainment or hitting function is defined (similarly to a cumulative distribution function) as

$$\begin{aligned} F(\vec{y}) &= \Pr[\mathcal{Y} \preceq_s \vec{y}] = \\ &= \Pr[\vec{Y}_1 \preceq \vec{y} \vee \vec{Y}_1 \preceq \vec{y} \vee \ldots \vee \vec{Y}_m \preceq \vec{y}] = \\ &= \Pr[\text{optimizer attains the goal } \vec{y} \text{ in a single run}] \end{aligned}$$

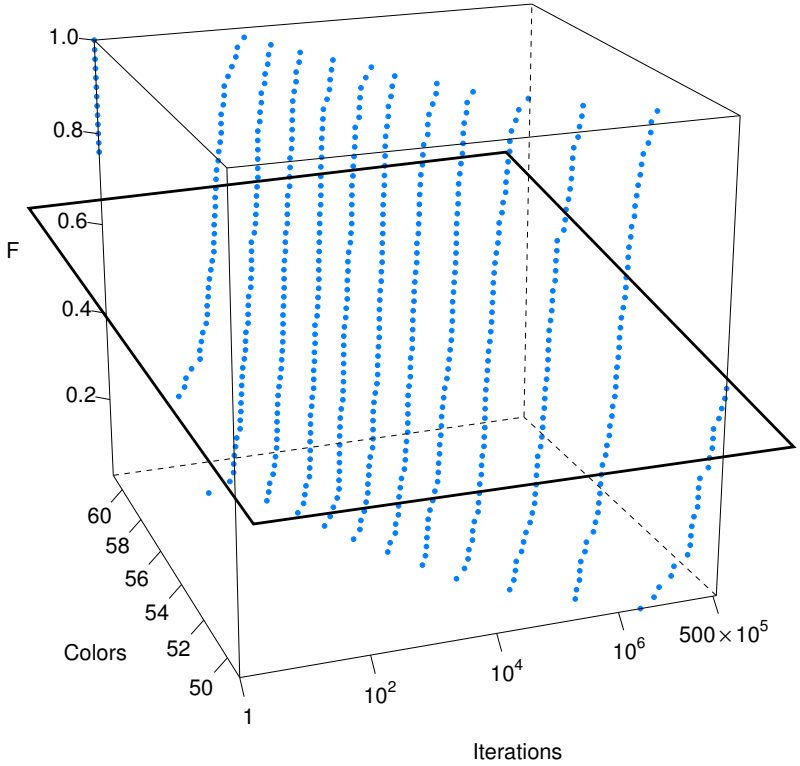
Sample data are collections of points $\mathcal{Y}_1, \ldots, \mathcal{Y}_n$ obtained by n independent runs. The corresponding ECDF is defined as:

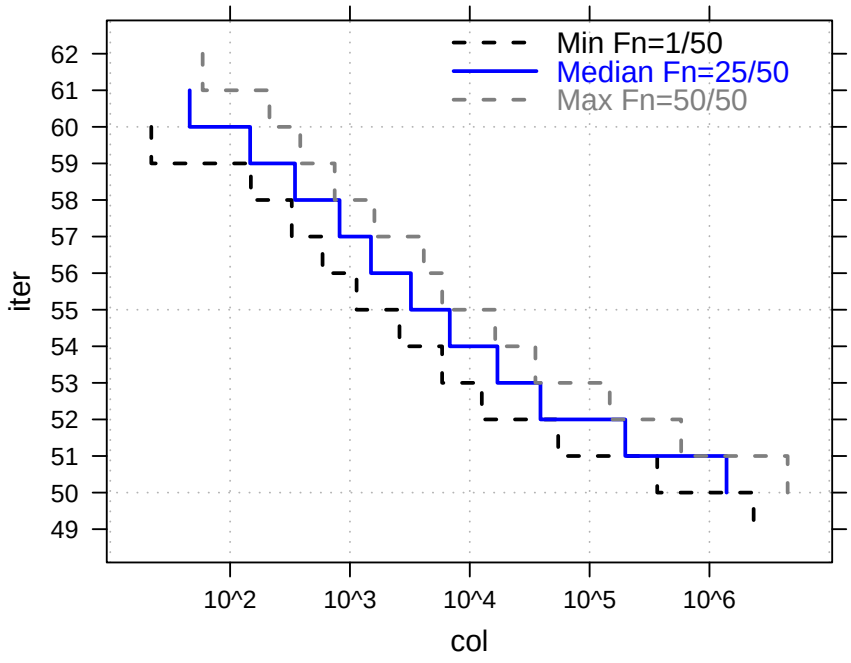
$$F_n(\vec{y}) = \frac{1}{n} \sum_{i=1}^{n} I(\mathcal{Y}_i \preceq_s \vec{y})$$

where $I(\mathcal{Y}_i \preceq_s \vec{y}) = 1$, if $\vec{Y}_1^i \preceq \vec{y}$ or $\vec{Y}_2^i \preceq \vec{y}$ or $\ldots$ or $\vec{Y}_{m_i}^i \preceq \vec{y}$.

Asymptotic Heuristics

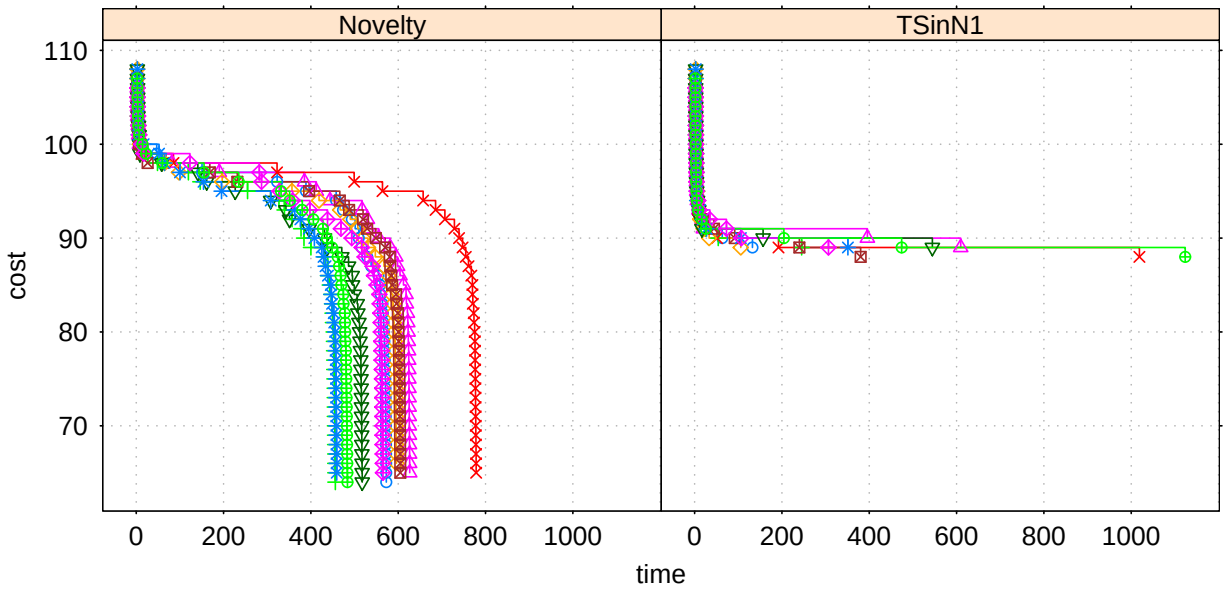
Approach 3, Bi-objective analysis

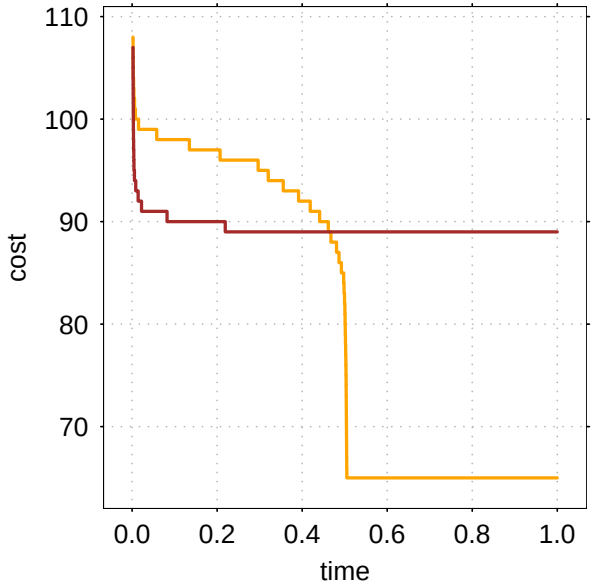




Asymptotic Heuristics

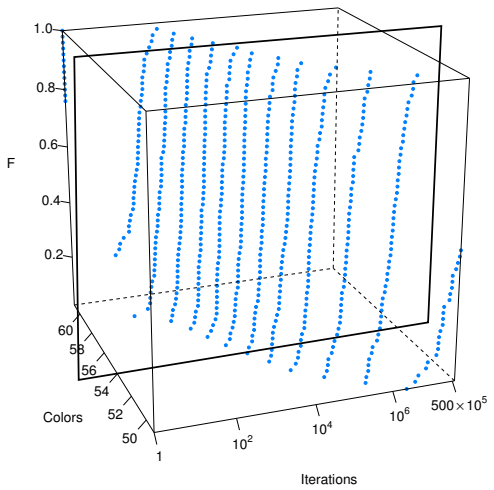
Approach 3, Bi-objective analysis: Example

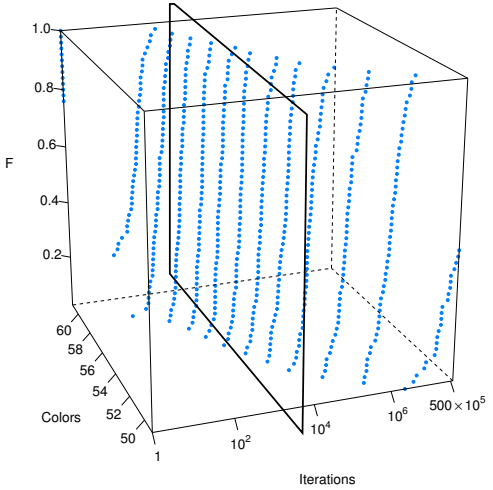


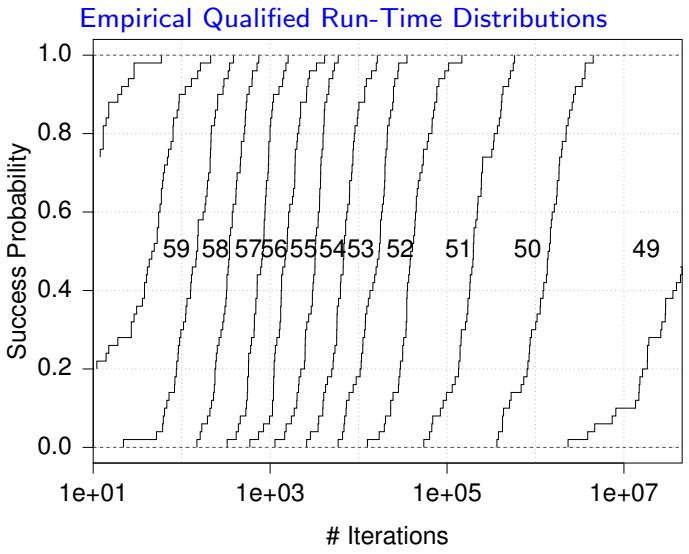


Asymptotic Heuristics

Approach 3, Bi-objective analysis

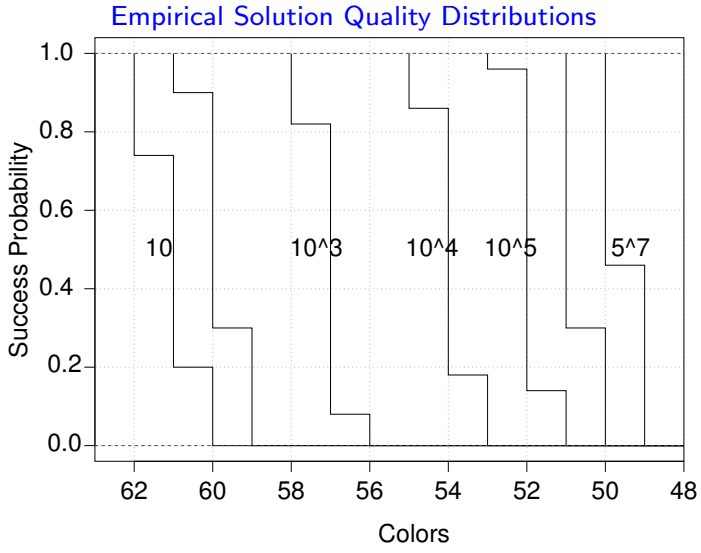






$$P(T \leq t \mid X \leq \overline{x}) = \frac{|\{i \mid \exists\, Y(X,T) \in \mathcal{Y}_i, X \leq \overline{x} \wedge T \leq t\}|}{|\{i \mid \exists\, Y(X,T) \in \mathcal{Y}_i, X \leq \overline{x}\}|}$$

$$P(X \leq x \mid T \leq \overline{t}) = \frac{|\{i \mid \exists\, Y(X,T) \in \mathcal{Y}_i, X \leq x \wedge T \leq \overline{t}\}|}{|\{i \mid \exists\, Y(X,T) \in \mathcal{Y}_i, T \leq \overline{t}\}|}$$



Outline

Correlation Analysis

Scenario:

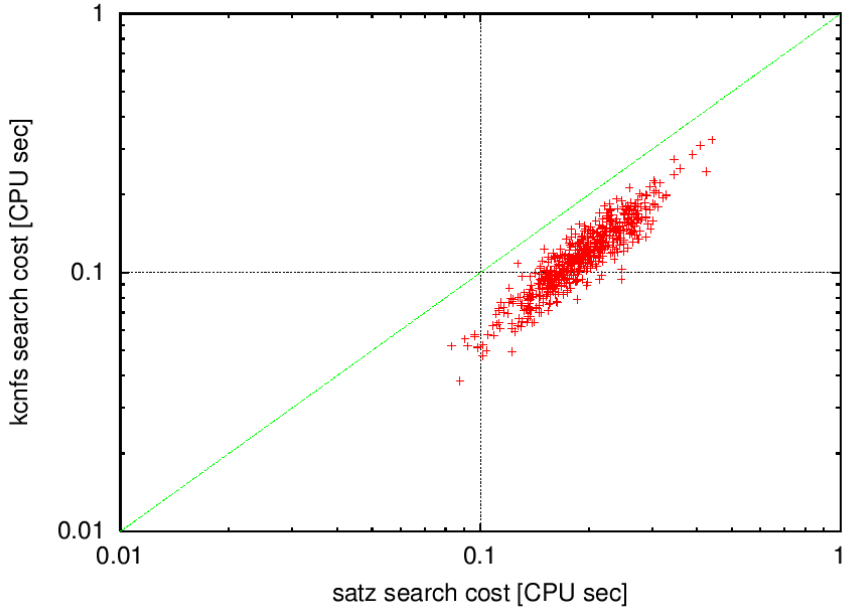
- ▷ heterogeneous instances, hence data transformation
- ▷ 1 or r run per instance
- ▷ consider time to goal or solution quality
- ▶ Interest: inspecting whether instances are all equally hard to solve for different algorithms or whether some features make the instances harder

Tools: correlation plots (each point represents an instance), correlation coefficient:
Population:

$$\rho_{XY} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y}$$

Sample:

$$r_{XY} = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)s_X s_Y}$$



[Hoos and Stützle, 2004]

Correlation Analysis

Scenario:

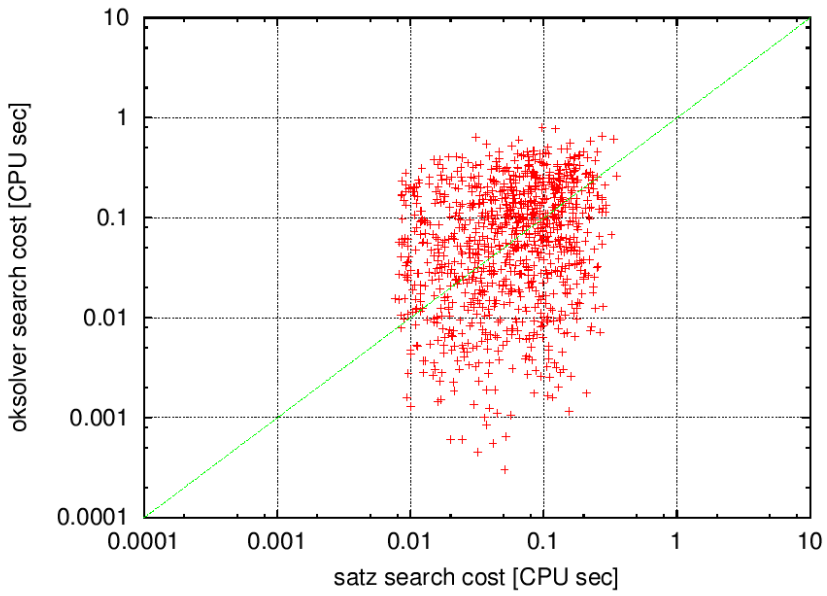
- ▷ heterogeneous instances, hence data transformation
- ▷ 1 or r run per instance
- ▷ consider time to goal or solution quality
- ▶ Interest: inspecting whether instances are all equally hard to solve for different algorithms or whether some features make the instances harder

Tools: correlation plots (each point represents an instance), correlation coefficient:
Population:

$$\rho_{XY} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y}$$

Sample:

$$r_{XY} = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)s_X s_Y}$$



[Hoos and Stützle, 2004]

Scaling Analysis

Scenario:

- ▷ one heuristic $\mathcal{A}^{\perp}$ or $\mathcal{A}^{\infty}$ with *a priori* quality goal
- ▷ data collected on instances of different size or features
- ▶ Interest: characterizing the growth of computational effort.

Given a set of data points (N_i, Y_i) obtained from an experiment in which $Y_i = f(N_i)$, for some unknown function $f(n)$,
Find growth function class $O(g_u(n))$ and/or $\Omega(g_l(n))$ to which $f(n)$ belongs.

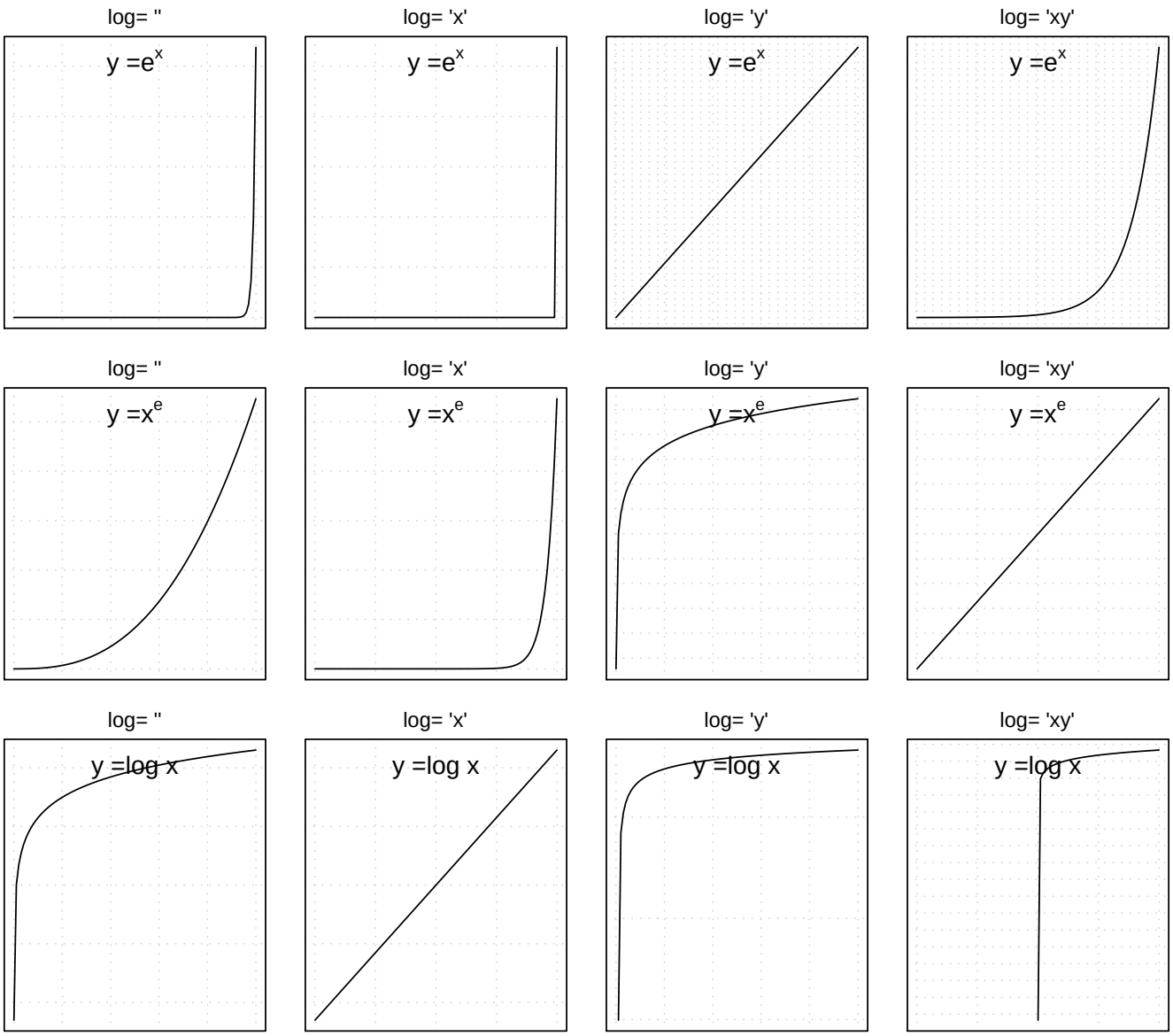
Mix of interpolation of the data trend and extrapolation beyond the range of experimentation

Tools:

- Heuristic adaptation of linear regression techniques
  - Log-log transformation and linear regression
  - Box-cox rules [McGeoch et al., 2002]
- Smoothing techniques

Scaling Analysis

Plots and Linear Trends



Linear Regression

Simple Linear Regression: *dependent variable* + *independent variable*

$$Y_i = \beta_0 + \beta X_i + \epsilon_i$$

Uses the Least Squares Method:

$$\min \sum_i \epsilon_i^2$$

$$\epsilon_i = Y_i - \beta_0 - \beta X_i$$

The indicator of the quality of fitness is the *coefficient of determination* R^2 (but use with caution)

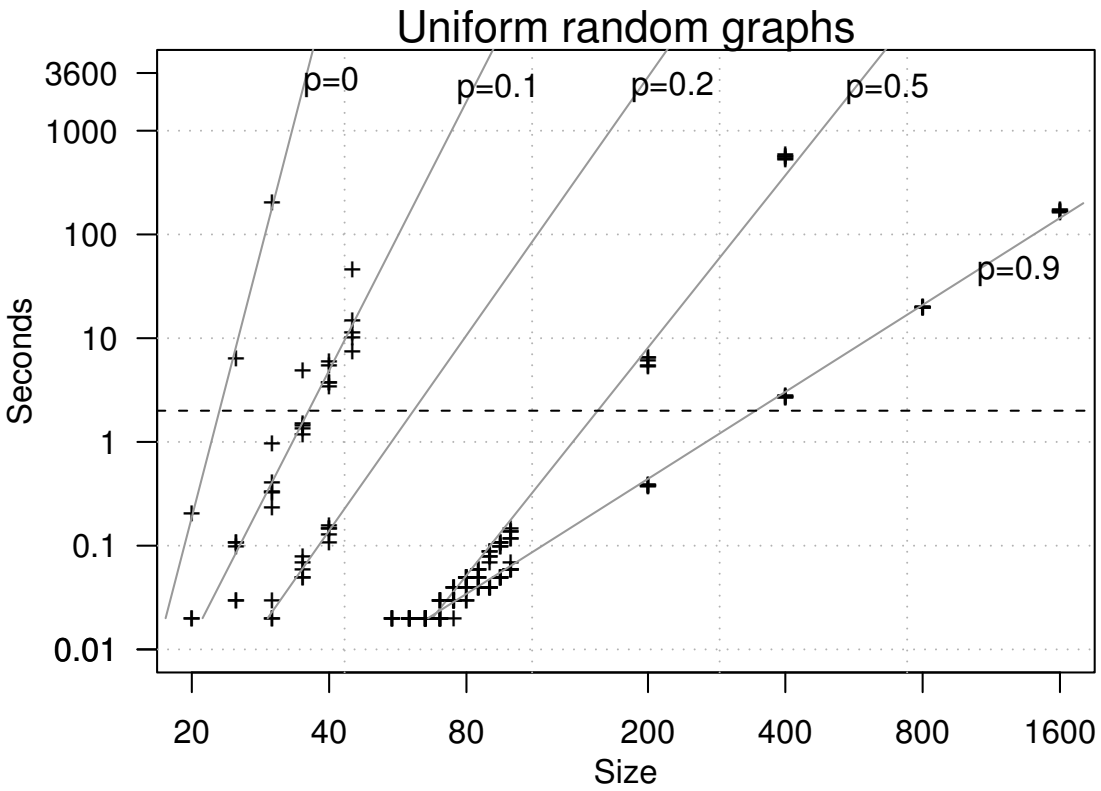
Multiple Linear regression considers multiple predictors

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \epsilon_i$$

The indicator of the quality of fitness is the *adjusted* R^2 statistic

Scaling Analysis
Example

Running time of RLF for Graph Coloring:
with two regressors: # vertices and edge density



Outline

Characterization of Run-time

Parametric models used in the analysis of run-times to:

- provide more informative experimental results
- make more statistically rigorous comparisons of algorithms
- exploit the properties of the model (eg, the character of long tails and completion rate)
- predict missing data in case of censored distributions

Procedure:

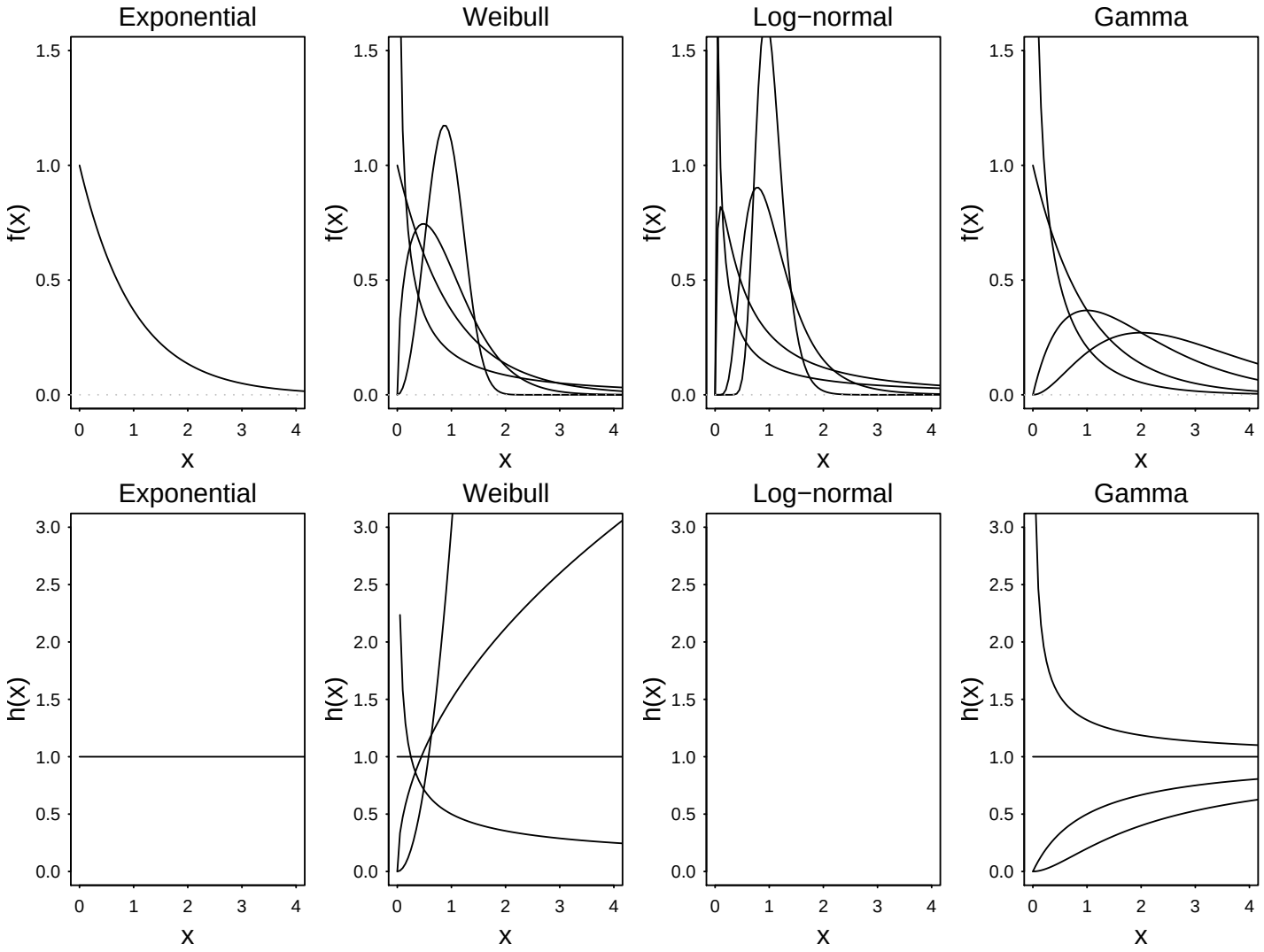
- choose a model
- apply fitting method
  maximum likelihood estimation method:

$$\max_{\theta \in \Theta} \log \prod_{i=1}^{n} p(X_i, \theta)$$

- test the model

Characterization of Run-time

The distributions used are [Frost et al., 1997; Gomes et al., 2000]:



Characterization of Run-time

Motivations for these distributions:

- qualitative information on the completion rate (= hazard function)
- empirical good fitting

To check whether a parametric family of models is reasonable the idea is to make plots that should be linear. Departures from linearity of the data can be easily appreciated by eye.

Example: for an exponential distribution:

$$\log S(t) = -\lambda t \qquad S(t) = 1 - F(t) \text{ is the survivor function}$$

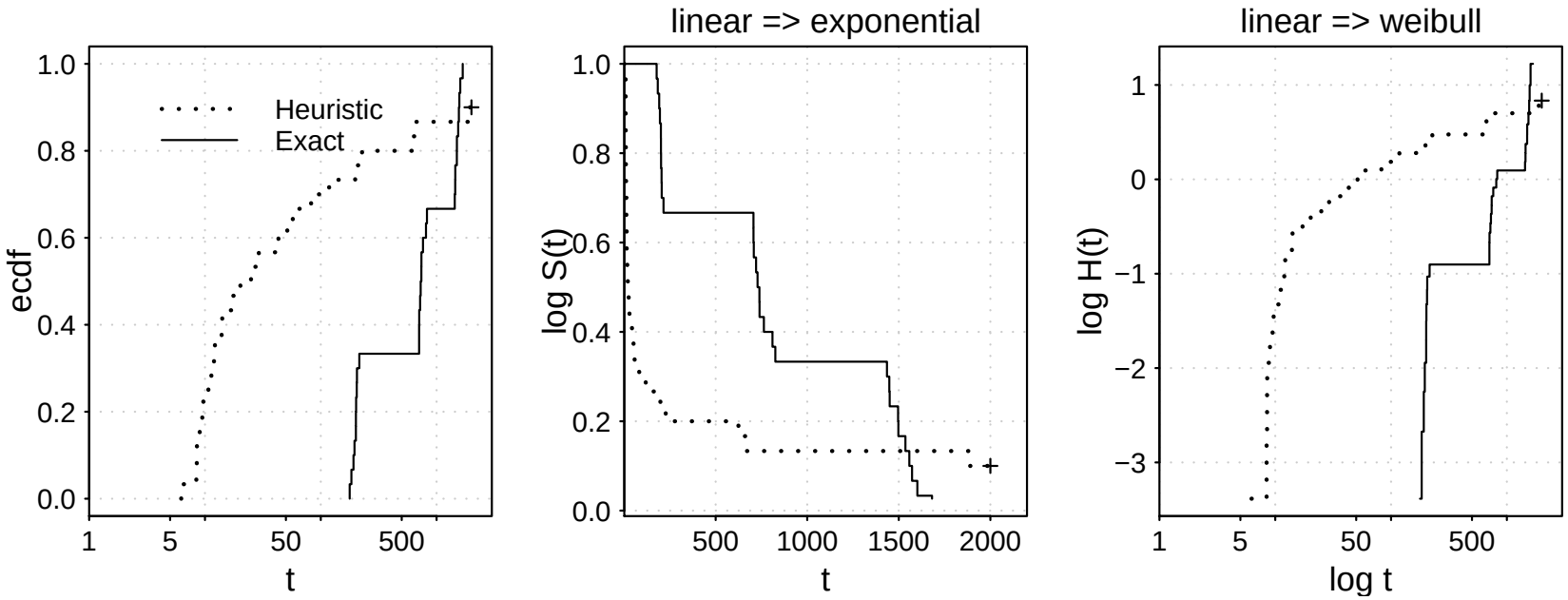
hence the plot of $\log S(t)$ against t should be linear

Similarly, for the Weibull the cumulative hazard function is linear on a log-log plot

Characterization of Run-time

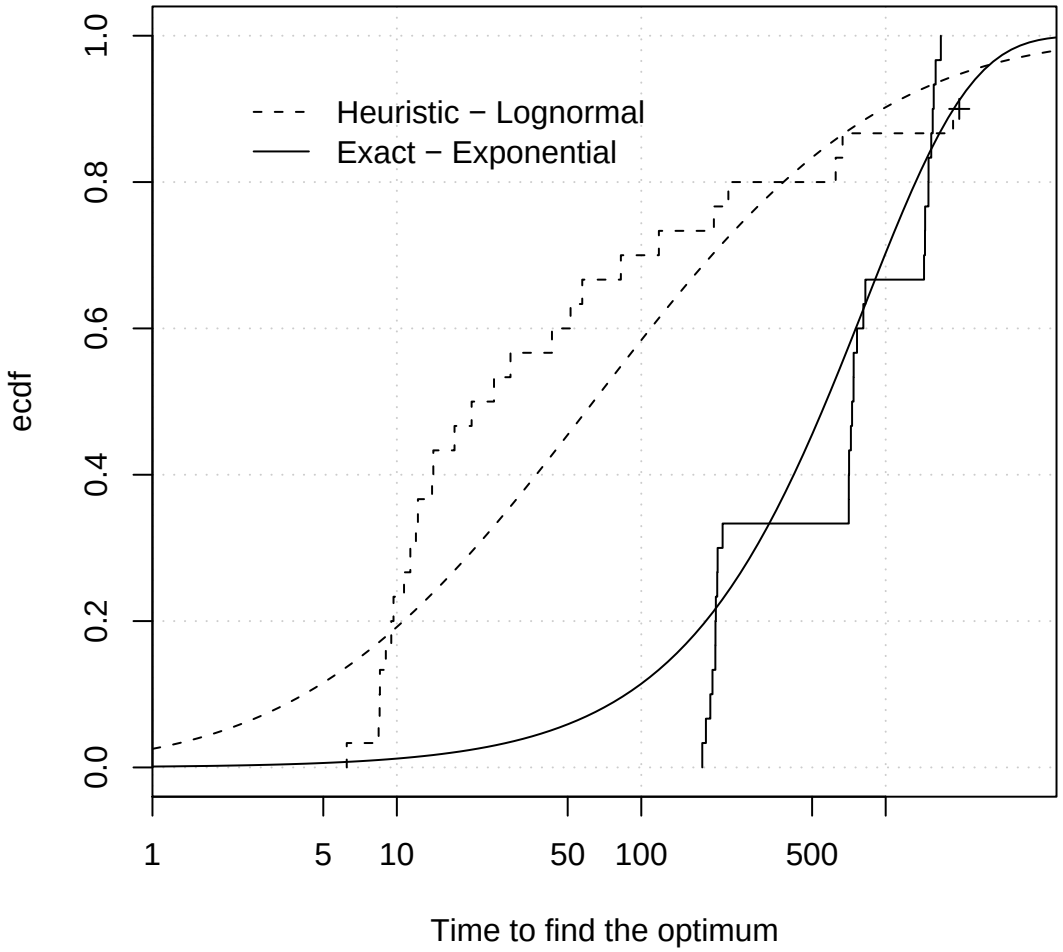
Example

Graphical inspection for the two censored distributions from the previous example on 2-edge-connectivity.



Characterization of Run-time

Example



Extreme Value Statistics

- Extreme value statistics focuses on characteristics related to the tails of a distribution function
  1. extreme quantiles (*e.g.*, minima)
  2. indices describing tail decay

- 'Classical' statistical theory: analysis of means.
  Central limit theorem: $X_1, \ldots, X_n$ i.i.d. with F_X

$$\sqrt{n}\frac{\bar{X} - \mu}{\sqrt{Var(X)}} \xrightarrow{D} N(0,1), \qquad \text{as } n \to \infty$$

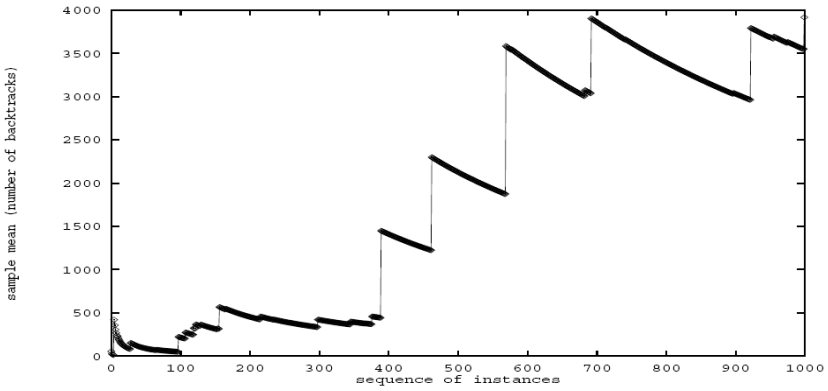
Heavy tailed distributions: mean and/or variance may not be finite!

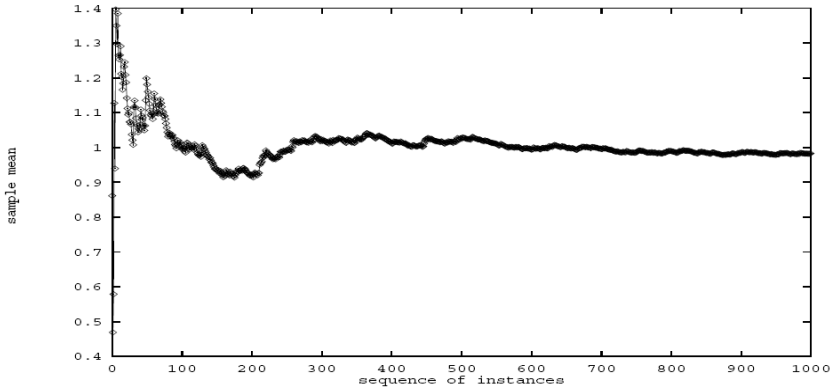
Characterization of Run-time

Heavy Tails

Gomes et al. [2000] analyze the mean computational cost to find a solution on a single instance

On the left, the observed behavior calculated over an increasing number of runs.
On the right, the case of data drawn from normal or gamma distributions





- The use of the median instead of the mean is recommended
- The existence of the moments (*e.g.*, mean, variance) is determined by the tails behavior: a case like the left one arises in presence of long tails

Outline

Inferential Statistics

- We work with samples (instances, solution quality)
- But we want sound conclusions: generalization over a given population (all possible instances)
- Thus we need statistical inference

$$\begin{array}{ccc} \text{Random Sample} & & \text{Population} \\ X^n & \xrightarrow{\text{Inference}} & P(x, \theta) \\ \text{Statistical Estimator } \widehat{\theta} & & \text{Parameter } \theta \end{array}$$

Since the analysis is based on finite-sized sampled data, statements like

> *"the cost of solutions returned by algorithm $\mathcal{A}$ is smaller than that of algorithm $\mathcal{B}$"*

must be completed by

> *"at a level of significance of 5%".*

A Motivating Example

- There is a competition and two algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ are submitted.
- We run both algorithms once on n instances.
  On each instance either $\mathcal{A}_1$ wins (+) or $\mathcal{A}_2$ wins (-) or they make a tie (=).

Questions:

1. If we have only 10 instances and algorithm $\mathcal{A}_1$ wins 7 times how confident are we in claiming that algorithm $\mathcal{A}_1$ is the best?
2. How many instances and how many wins should we observe to gain a confidence of 95% that the algorithm $\mathcal{A}_1$ is the best?

A Motivating Example

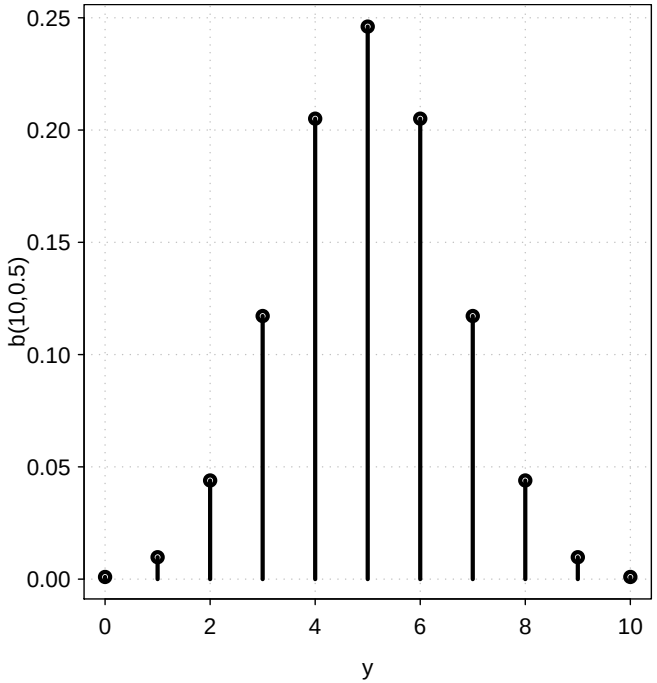
- p: probability that $\mathcal{A}_1$ wins on each instance $(+)$
- n: number of runs without ties
- Y: number of wins of algorithm $\mathcal{A}_1$

If each run is indepenedent and consitent:

$$Y \sim b(n, p): \qquad \Pr[Y = y] = \binom{n}{y} p^y (1-p)^{1-y}$$

Under the conditions of question 1, we can check how unlikely the situation is if it were $p(+) \leq p(-)$.

If $p = 0.5$ then the chance that algorithm $\mathcal{A}_1$ wins 7 or more times out of 10 is 17.2%: quite high!



A Motivating Example

- p: probability that $\mathcal{A}_1$ wins on each instance $(+)$
- n: number of runs without ties
- Y: number of wins of algorithm $\mathcal{A}_1$

If each run is indepenedent and consitent:

$$Y \sim b(n, p): \qquad \Pr[Y = y] = \binom{n}{y} p^y (1-p)^{1-y}$$

Under the conditions of question 1, we can check how unlikely the situation is if it were $p(+) \leq p(-)$.

If $p = 0.5$ then the chance that algorithm $\mathcal{A}_1$ wins 7 or more times out of 10 is 17.2%: quite high!

To answer question 2. we compute the 95% quantile, *i.e.*, $y : \Pr[Y \geq y] < 0.05$ with $p = 0.5$ at different values of n:

n	10	11	12	13	14	15
y	9	9	10	10	11	12

n	16	17	18	19	20
y	12	13	13	14	15

Summary

What we saw:

- Simple Comparative Analysis
- Characterization

What remains:

- Methods for statistical inference
- Experimental Designs techniques for component based analysis
- Advanced Designs for tuning and configuring

(This will be covered in the next talks and in Ruben's tutorial)

But also:

- Reactive/Learning approaches
- Phase transitions
- Landscape analysis
- ...

References

H. Hoos, T. Stützle (2004). *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann Publishers, San Francisco, CA, USA.

C. Demetrescu, G. F. Italiano (2000). What do we learn from experimental algorithmics? In M. Nielsen, B. Rovan (Eds.), *MFCS*, vol. 1893 of *Lecture Notes in Computer Science*, pp. 36–51, Springer Verlag, Berlin, Germany, Berlin.

C. C. McGeoch (1996). Toward an experimental method for algorithm simulation. *INFORMS Journal on Computing*, vol. 8, no. 1, pp. 1–15.

E. Zemel (August 1981). Measuring the quality of approximate solutions to zero-one programming problems. *Mathematics of operations research*, vol. 6, no. 3, pp. 319–332.

V. G. da Fonseca, C. M. Fonseca, A. O. Hall (2001). Inferential performance assessment of stochastic optimisers and the attainment function. In C. Coello, D. Corne, K. Deb, L. Thiele, E. Zitzler (Eds.), *Proceedings of Evolutionary Multi-Criterion Optimization First International Conference, EMO 2001*, vol. 1993 of *Lecture Notes in Computer Science*, pp. 213–224, Springer Verlag, Berlin, Germany.

E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, V. G. da Fonseca (2003). Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Trans. Evolutionary Computation*, vol. 7, no. 2, pp. 117–132.

C. McGeoch, P. Sanders, R. Fleischer, P. R. Cohen, D. Precup (2002). Using finite experiments to study asymptotic performance. In R. Fleischer, B. Moret, E. M. Schmidt (Eds.), *Experimental Algorithmics: From Algorithm Design to Robust and Efficient Software*, vol. 2547 of *Lecture Notes in Computer Science*, pp. 93–126, Springer Verlag, Berlin, Germany.

D. Frost, I. Rish, L. Vila (1997). Summarizing csp hardness with continuous probability distributions. In *AAAI/IAAI*, pp. 327–333.

C. Gomes, B. Selman, N. Crato, H. Kautz (2000). Heavy-tailed phenomena in satisfiability and constraint satisfaction problems. *Journal of Automated Reasoning*, vol. 24, no. 1-2, pp. 67–100.

I. M. Ovacik, S. Rajagopalan, R. Uzsoy (2000). Integrating interval estimates of global optima and local search methods for combinatorial optimization problems. *Journal of Heuristics*, vol. 6, no. 4, pp. 481–500.

J. Hüsler, P. Cruz, A. Hall, C. M. Fonseca (2003). On optimization and extreme value theory. *Methodology and Computing in Applied Probability*, vol. 5, pp. 183–195.

M. Birattari (2005). *The Problem of Tuning Metaheuristics as Seen from a Machine Learning Perspective*. No. DISKI 292, Infix/Aka, Berlin, Germany.

J. D. Petruccelli, B. Nandram, M. Chen (1999). *Applied Statistics for Engineers and Scientists*. Prentice Hall, Englewood Cliffs, NJ, USA.

W. Conover (1999). *Practical Nonparametric Statistics*. John Wiley & Sons, New York, NY, USA, 3rd edn.

D. C. Montgomery, G. C. Runger (2007). *Applied Statistics and Probability for Engineers*. John Wiley & Sons, forth edn.

J. F. Lawless (1982). *Statistical Models and Methods for Lifetime Data*. Wiley Series in Probability and Mathematical Statistics, jws.

G. Seber (2004). *Multivariate observations*. Wiley series in probability and statistics, John Wiley.

M. H. Kutner, C. J. Nachtsheim, J. Neter, W. Li (2005). *Applied Linear Statistical Models*. McGraw Hill, 5th edn.