# Parametric statistics for program performance analysis, comparison and evaluation

Julien Worms and Sid Touati

HPCS, Genova, July 2017











## Observed Program Performance Variability

- Fixed binary program and a fixed machine with a fixed software environment with fixed data input.
- Repeat the program execution $n$ times.
- You get $n$ distinct performance numbers.
- If you do not observe variability, you do not need statistics.





## Why program performances vary ?

- Physical technology
  - Variable CPU frequency, asynchronous peripherals, input/output.

## Why program performances vary ?

- Physical technology
  - Variable CPU frequency, asynchronous peripherals, input/output.
- Processor micro-architecture designs
  - OoO execution, branch prediction, data pre-fetching, memory hierarchy, etc.





## Why program performances vary ?

- Physical technology
  - Variable CPU frequency, asynchronous peripherals, input/output.
- Processor micro-architecture designs
  - OoO execution, branch prediction, data pre-fetching, memory hierarchy, etc.
- Competition between peripherals
  - Shared memory, NUMA, communication network.





## Why program performances vary ?

- Physical technology
  - Variable CPU frequency, asynchronous peripherals, input/output.
- Processor micro-architecture designs
  - OoO execution, branch prediction, data pre-fetching, memory hierarchy, etc.
- Competition between peripherals
  - Shared memory, NUMA, communication network.
- Operating systems
  - Virtual memory management, process memory layout, thread and process scheduling.

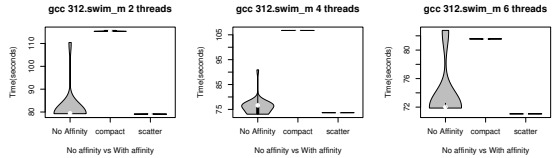




## Why program performances vary ?

- Physical technology
  - Variable CPU frequency, asynchronous peripherals, input/output.
- Processor micro-architecture designs
  - OoO execution, branch prediction, data pre-fetching, memory hierarchy, etc.
- Competition between peripherals
  - Shared memory, NUMA, communication network.
- Operating systems
  - Virtual memory management, process memory layout, thread and process scheduling.
- Algorithmic factors
  - Parallel programming with imbalanced workload.
  - Non deterministic algorithms.

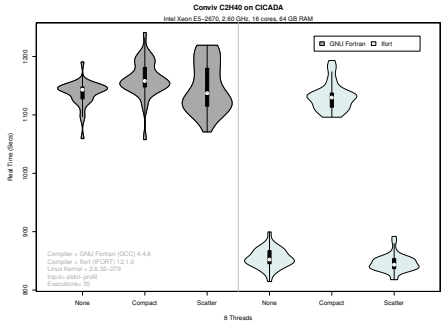
## Example of experiences on a dedicated machine








## Example of experience on a shared machine with dedicated CPU








## Before doing statistics: basic assumptions

- Independence of the measurements.
- Continuous model vs. discrete model.





## Parametric vs. non-parametric statistics

- Parametric statistics: assume a p.d.f.
  - Ex: Student t-test.
- Non-Parametric statistics: any kind of data distribution.
  - Ex: Wilcoxon-Mann-Whitney, chi-square test, central limit theorem.

## Advantages of parametric statistics

- Parameters of a parametric model can often be interpreted.
- Formal mathematical proofs.
- More accurate for large data sets and multi-dimensional data.





## Advantages of non-parametric statistics

- Do not need a mathematical model.
- Do not need to build new statistics for every data distribution.
- For large data sets, it works pretty well (but without proof).





## Standard program performance metrics

- Summarise $n$ performance numbers by a single one:
  - Mean
  - Median
  - Minimum
  - Maximum
- You lose information.

How to model precisely the performances of a program to be able to take reliable conclusions/decisions ?





## Are the data distributions of a Gaussian nature ?

- 10 years of collected data performances
- Various machines architectures and configurations.
- SPEC CPU applications (2001, 2006), all SPEC OMP applications, NAS Parallel Benchmark, own micro-benchmarks, other parallel applications, various compilers versions and options, Linux versions.
- 2438 data samples, each one contains between 30 and 1000 execution times.
- We did a normality test (Shapiro).
- With a risk of 5%, 67% of the samples are not of Gaussian nature.

## Standard tests in statistics: Speedup-Test for program performances

- $t$-test of Student to compare between two theoretical means.
  - Program performances do not follow normal distributions in general.
- Wilcoxon-Mann-Whitney to compare between two theoretical medians.
  - Is a single median a good summary of $n$ data ?





## Which function can we use to model a continuous random variable ?

Some facts

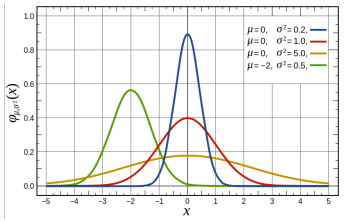
- Every sample has an infinite possible theoretical models.
- In the past, some density functions have been selected for their mathematical characteristics (to ease formal problem solving by hand).
- It is impossible to formally prove that one model is better than another, the theoretical distribution remains always unknown.
- Statistics do not provide guarantees, they help decision and analysis of complex/random phenomena.





## Gaussian law

Used in physics

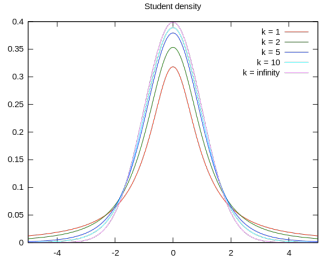






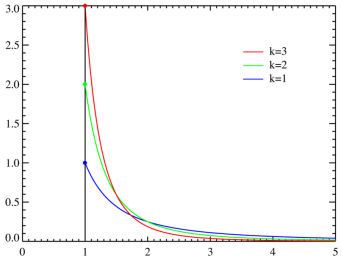

## Student law

Used to compute the confidence interval of the average



## Pareto law

Used in queue theory, quality management, etc.

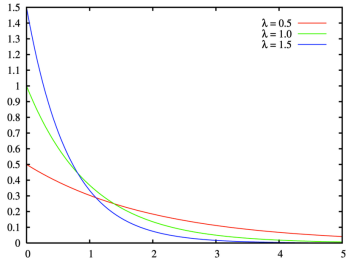







## Exponential law

Used in electronics and in radioactivty.

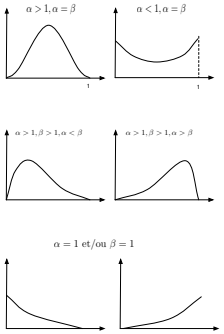







## Gamma law

It is a generalisation of other laws.

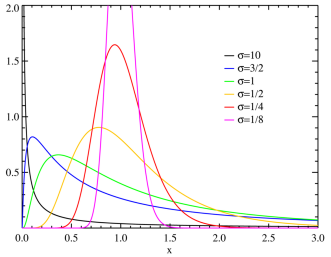






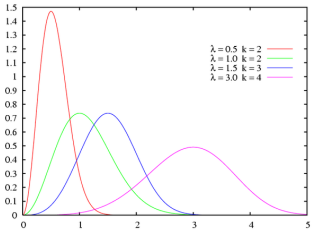

## Lognormal law

Used in finance, stock market, and many other domains.



## Weibull law

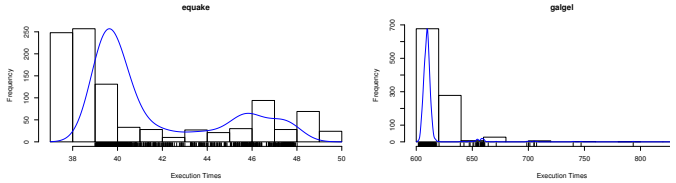
Used in the science of materials








## Many programs performances follow multi-modal distributions








## Gaussian mixture

The program performances is modelled with a probability density function equal to the sum of $K$ gaussians:

$$f_X(u) = \sum_{k=1}^{K} \pi_k . \varphi_X(\mu_k, \sigma_k, u)$$

where $\sum_{k=1}^{K} \pi_k = 1$ and $\varphi_X(\mu_k, \sigma_k)$ is the probability density function of a gaussian with mean $\mu_k$ et standard deviation $\sigma_k$.

$$\varphi_X(\mu, \sigma, u) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{u-\mu}{\sigma}\right)^2}$$





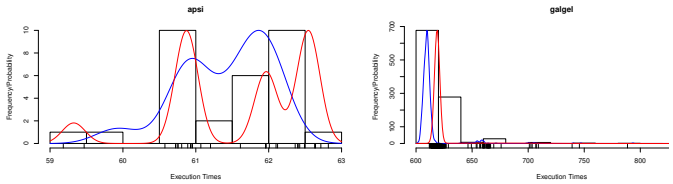
## Clustering in statistics

Given a data sample of size $n$, compute estimators of:

- The number $K$ of gaussians (called *clusters*);
- The weight $\pi_k$ of each cluster $k$;
- The theoretical mean $\mu_k$ of each cluster $k$;
- The standard deviation $\sigma_k$ of each cluster $k$.

A famous algorithm called *EM* does the job.

## Examples of clustering

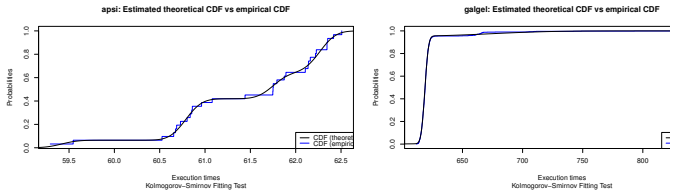







## Checking the fitting of the data to the GM (Kolmogorov-Smirnov test)

We designed a KS test adequately calibrated with bootstrap.








## Clustering results

with a risk error of 5%

- 83% of the samples are modelled correctly with GM.
- 17% of the samples are rejected.





## Why 17% of the sample are rejected ?

- *ties* (identical values), rounding errors when collecting performance data.
  - this problem is easily fixed by increasing the precision of the measurements.
- Heavy-tail distributions cannot be approximated easily with GM.

## New performance metrics

Let $X$ and $Y$ be two random variables representing the performances of two programs

1. The mean difference: $\mathcal{I}_1 = \mathbb{E}[|X - Y|]$
2. The probability that a single program run is better than another: $\mathcal{I}_2 = \mathbb{P}[X < Y]$.
3. The probability that a single run is better than all the others: $\mathcal{I}_3 = \mathbb{P}[X_1 < \min(X_2, \cdots, X_r)] = \mathbb{E}\left[\mathbb{1}_{X_1 < \min(X_2, \cdots, X_r)}\right]$
4. The variability level $\mathcal{I}_4$: the number of *modes*.





## Metric 1: $\mathbb{E}[|X - Y|]$

$\mathcal{X}$ is a sample of size $n$, $\mathcal{Y}$ is a samble of size $m$. $X$ et $Y$ are approximated by GM $(K, \mu, \sigma)$ and $(K', \mu', \sigma')$ resp.

1. Non-parametric estimation: $\mathbb{E}[|X - Y|] = \frac{1}{nm} \sum_{i=1,n} \sum_{j_1,m} |x_i - y_j|$
2. Parametric estimation with GM: $\mathbb{E}[|X - Y|] =$

$$\sum_{i=1}^{K} \sum_{j=1}^{K'} \pi_i \pi'_j$$

$$\left( (\mu_i - \mu'_j) \left( 1 - 2\Phi\left(\frac{\mu'^j - \mu^i}{\sqrt{\sigma^2 + \sigma'^2}}\right)\right) + \sqrt{\frac{2(\sigma_i^2 + \sigma_i^2)}{\pi}} e^{-\frac{(\mu_i - \mu'_j)^2}{2(\sigma_i^2 + \sigma'^2)}} \right)$$

where: $\Phi$ is the CDF of the Gaussian $\varphi(0, 1, u)$





## Metric 2: $\mathbb{P}[X < Y]$

$\mathcal{X}$ is a sample of size $n$, $\mathcal{Y}$ is a samble of size $m$. $X$ et $Y$ are approximated by GM $(K, \mu, \sigma)$ and $(K', \mu', \sigma')$ resp.

1. Non-parametric estimation (Man-Whitney statistic): $\mathbb{P}[X < Y] = \mathbb{E}[\mathbb{1}_{X<Y}] = \frac{1}{nm} \sum_{i=1,n} \sum_{j_1,m} \mathbb{1}_{x_i < y_j}$
2. Parametric estimation with GM: $\mathbb{P}[X < Y] = \sum_{i=1}^{K} \sum_{j=1}^{K'} \pi_i \pi'_j \Phi\left(\frac{\mu'^j - \mu^i}{\sqrt{\sigma^2 + \sigma'^2}}\right)$
   where: $\Phi$ is the CDF of the Gaussian $\varphi(0, 1, u)$





## Metric 3: $\mathbb{P}[X_1 \leq \min(X_2, \cdots, X_m)]$

The computation is done numerically not symbolically.

1. Non-parametric estimation

$$\mathbb{P}[X_1 \leq \min(X_2, \cdots, X_m)] =$$

$$\frac{1}{\prod_{j=1}^{r} n_j} \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_r=1}^{n_r} \mathbb{1}_{x_{i_1,1} < \min(x_{i_2,2}, \cdots, x_{i_r,r})}$$

2. Parametric estimation. We note $Y = \min(X_2, \cdots, X_m)$ and $G$ its CDF. $\mathbb{P}[X < Y] = \mathbb{E}[1 - G(x)] = \int (1 - G(u)) f_1(u) du$ where $f_1(u) = \sum_{i=1}^{K_1} \frac{\pi_{1i}}{\sigma_{1i}} \varphi\left(\frac{u - \mu_{1k}}{\sigma 1k}\right)$

## Metric 4: Variability level

- The variance is a measure of dispersion around the average, not easy to interpret !

$$\mathbb{P}\left[|X-\mu| > 4\sigma_X\right] \text{is very small}$$

- Our proposal: variability level= number of modes.





## Empirical study of variability levels of programs execution times:

The variability level: the number of *modes*.

- $\approx$ 37% of the samples have a variability level equal to 1.
- $\approx$ 32% of the samples have a variability level equal to 2.
- $\approx$ 12% of the samples have a variability level equal to 3.
- $\approx$ 19% of the samples have a variability level $\geq 4$.





## Limitations

- Some cases (Heavy-tail distributions) cannot be well modelled with GM.
- GM are not appropriate models for studying extreme value performances (minimum and maximum).





## Future plan

- Consider multi-dimensional performance data: tuples of values of distinct nature {execution time, energy consumption, memory consumption, network traffic, etc}.
- It is very difficult to have satisfactory regression models for multi-dimensional data.
- GM models are well adapted for such mathematical modelling.

## Conclusions

- Observed program performances are multi-modal;
- Modelling program performances with gaussian mixtures;
- Test of the fitting between the GM model and the data;
- New performance metrics;
- Free software called *VARCORE*.





## Main reference

Full research report (70 pages) with free software and demo.
*Parametric and Non-Parametric Statistics for Program Performance Analysis and Comparison*.
https://hal.inria.fr/hal-01286112





## VARCORE software

- Programmed with R.
- Requires free packages: mclust, R.utils
- Documented.





## Example 1: analysing the performances of one program

- Load performance data of one program
- Apply clustering (build a GM model)
- Check if the GM model is good enough.
- Compute the variability level.
- Print and plot the results.

## Example 2: comparing the performances of multiple codes versions

- Load performance data of multiple codes versions.
- Apply clustering (build a GM model) for each one.
- Check if the GM model is good enough for each one.
- Decide which is the best code version.
- Compute other performance metrics.