Extra-P:

Insightful Automatic Performance Modeling

Alexandru Calotoiu[1], Torsten Hoefler[2], Martin Schulz[3]
Sergei Shudler[1] and Felix Wolf[1]

[1] TU Darmstadt , [2] ETH Zürich , [3] TU München

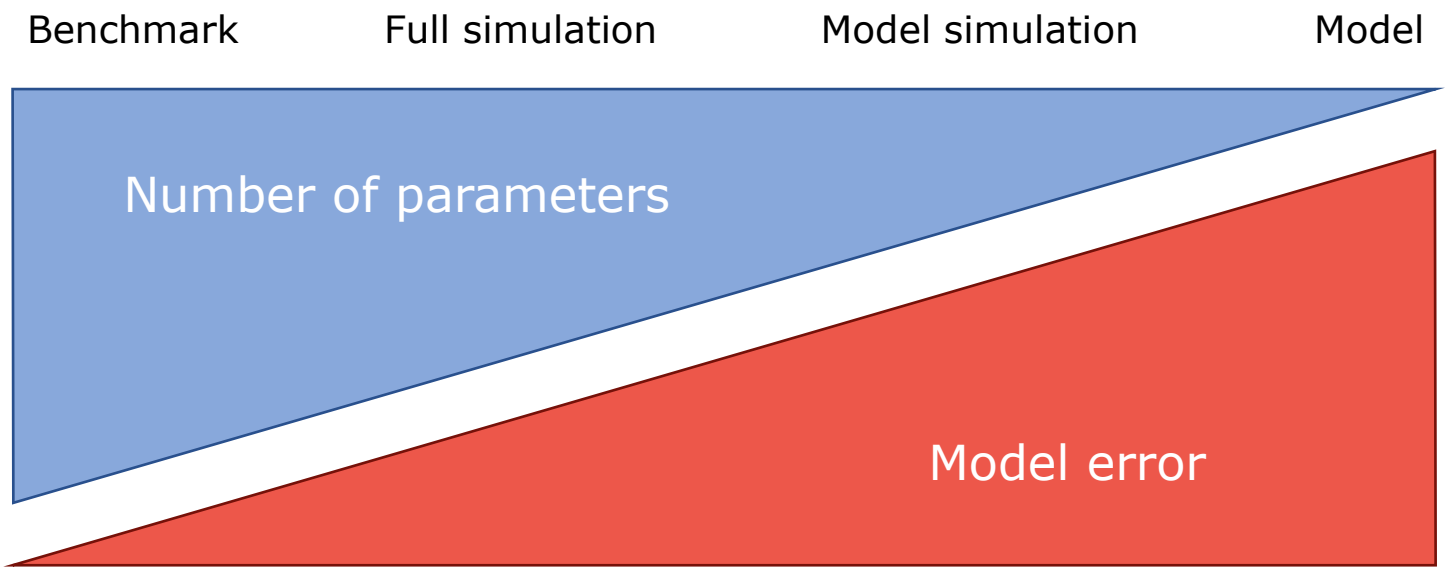
TECHNISCHE UNIVERSITÄT DARMSTADT

ETH zürich

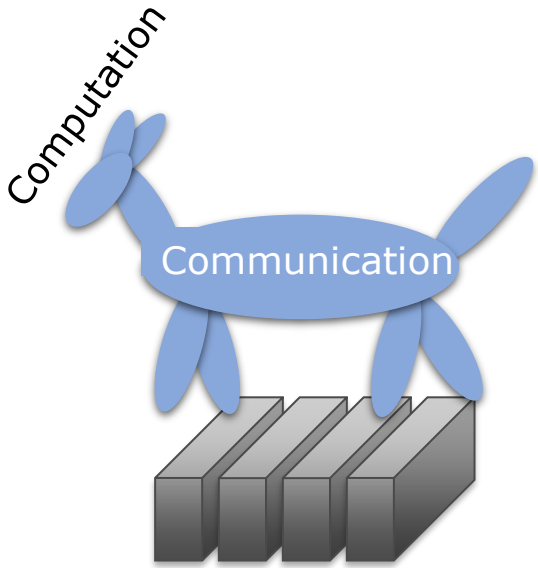
Lawrence Livermore National Laboratory

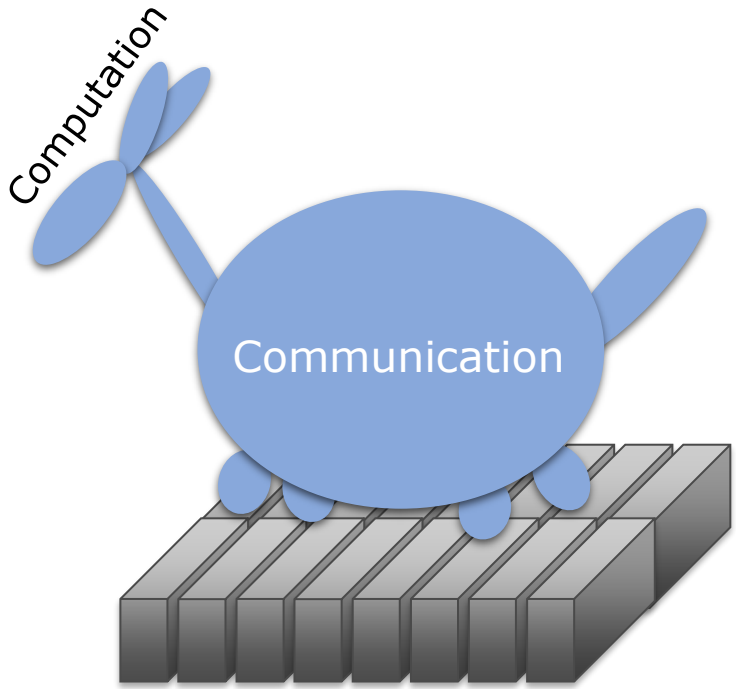
Technische Universität München

Spectrum of performance analysis methods

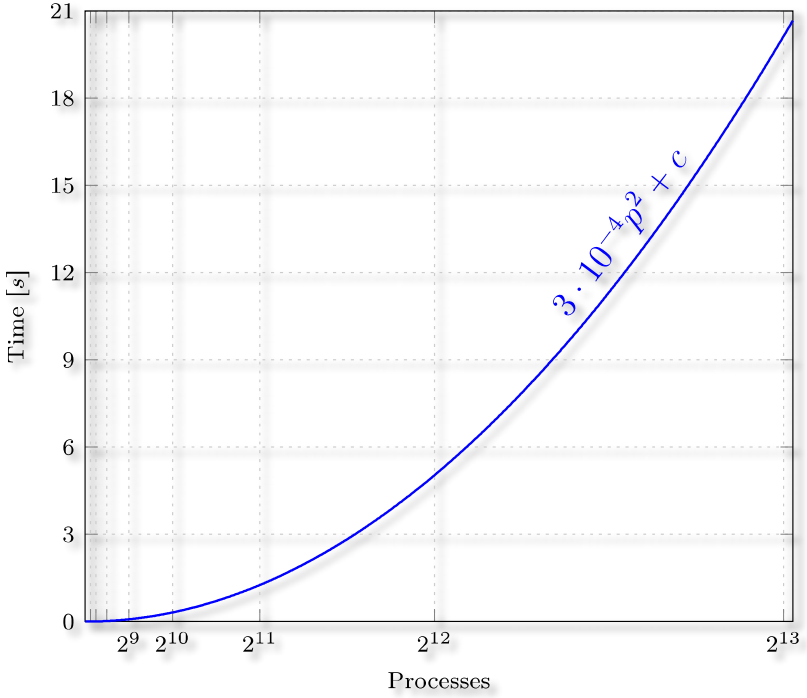


Motivation - latent scalability bugs





Scaling model



- Represents performance metric as a function of the number of processes
- Provides insight into the program behavior at scale

Analytical performance modeling

Identify kernels

- Parts of the program that dominate its performance at larger scales
- Identified via small-scale tests and intuition

Create models

- Laborious process
- Still confined to a small community of skilled experts

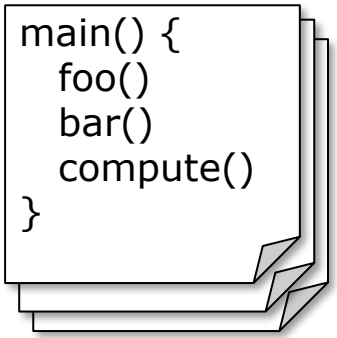
Disadvantages:

- Time consuming
- Danger of overlooking unscalable code

Hoisie et al.: *Performance and scalability analysis of teraflop-scale parallel architectures using multi-dimensional wavefront applications.* International Journal of High Performance Computing Applications,2000

Bauer et al.: *Analysis of the MILC Lattice QCD Application su3_rmd*. CCGrid, 2012

Automatic performance modeling

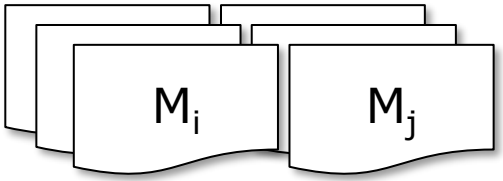


Input

Instrumentation

- All functions

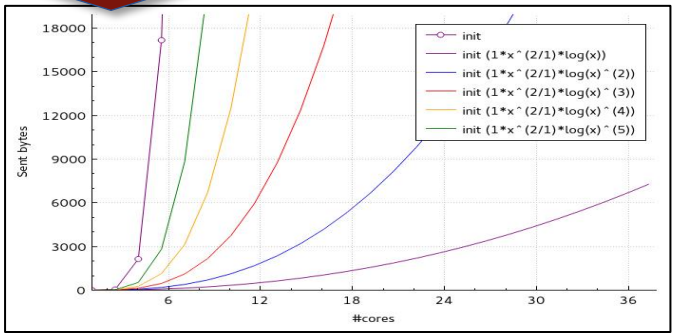
Performance measurements



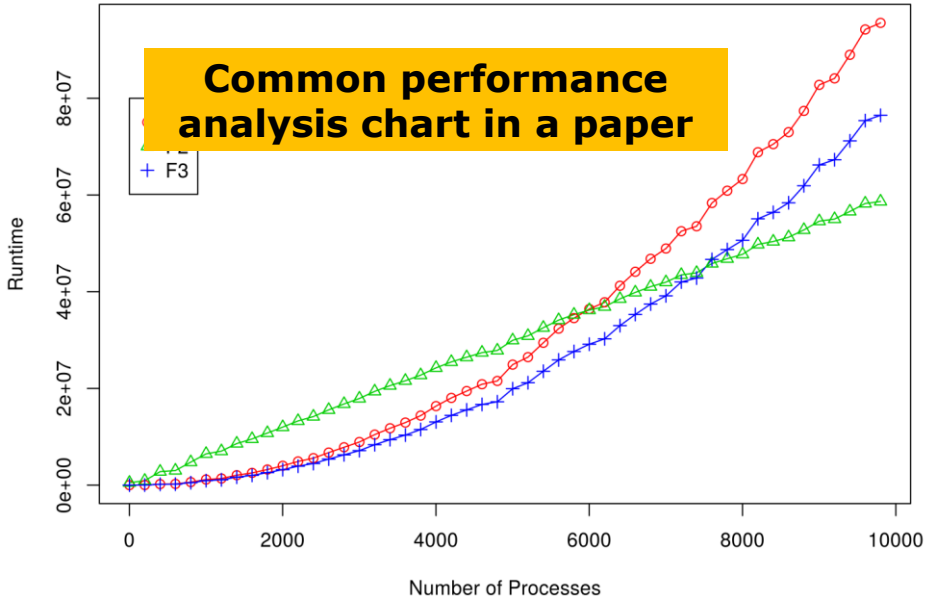
Extra-P

Output

Human-readable performance models of all functions (e.g., $t = c_1 * \log(p) + c_2$)



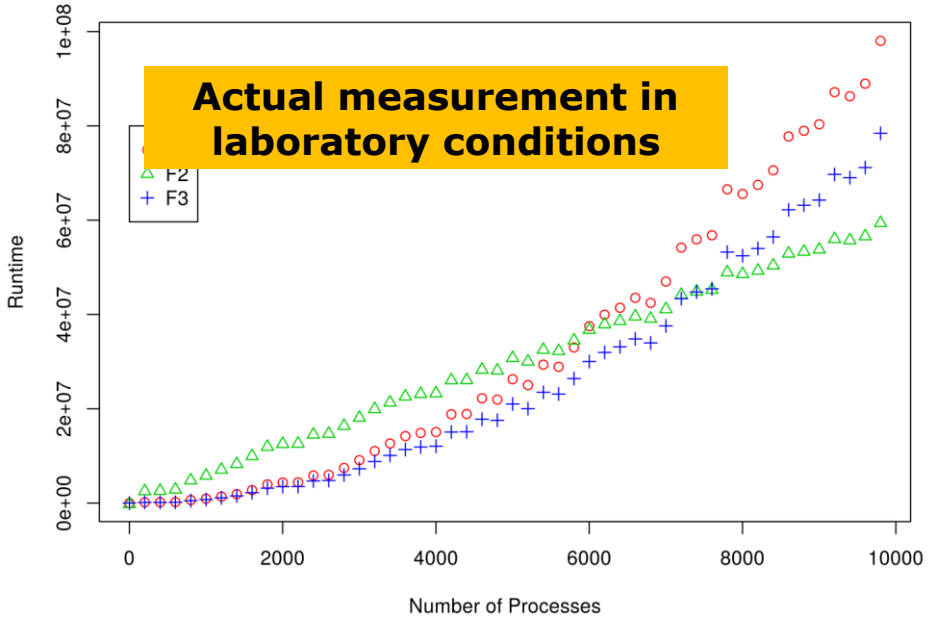
Primary focus on scaling trend



Ranking

1. F_2
2. F_1
3. F_3

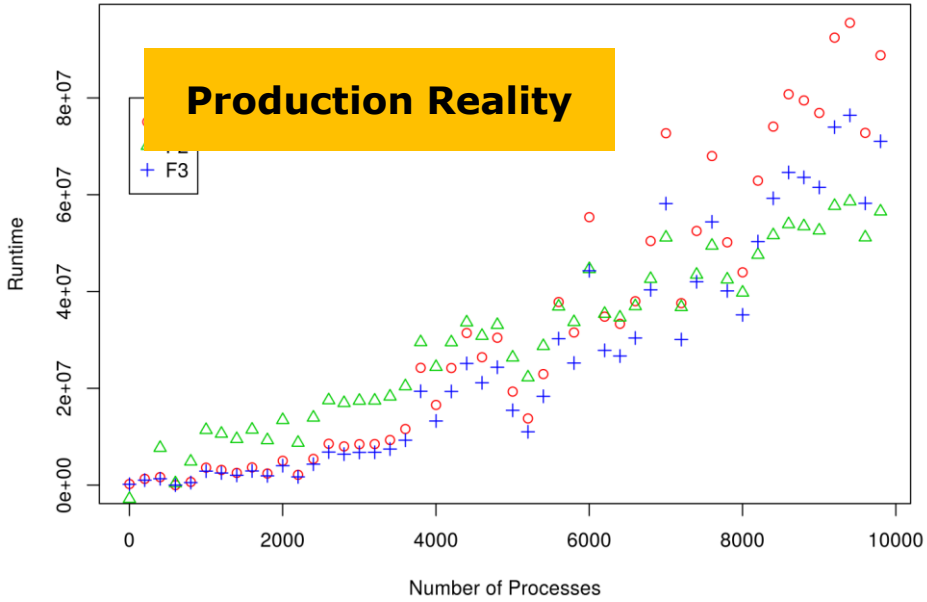
Primary focus on scaling trend



Ranking

1. F_2
2. F_1
3. F_3

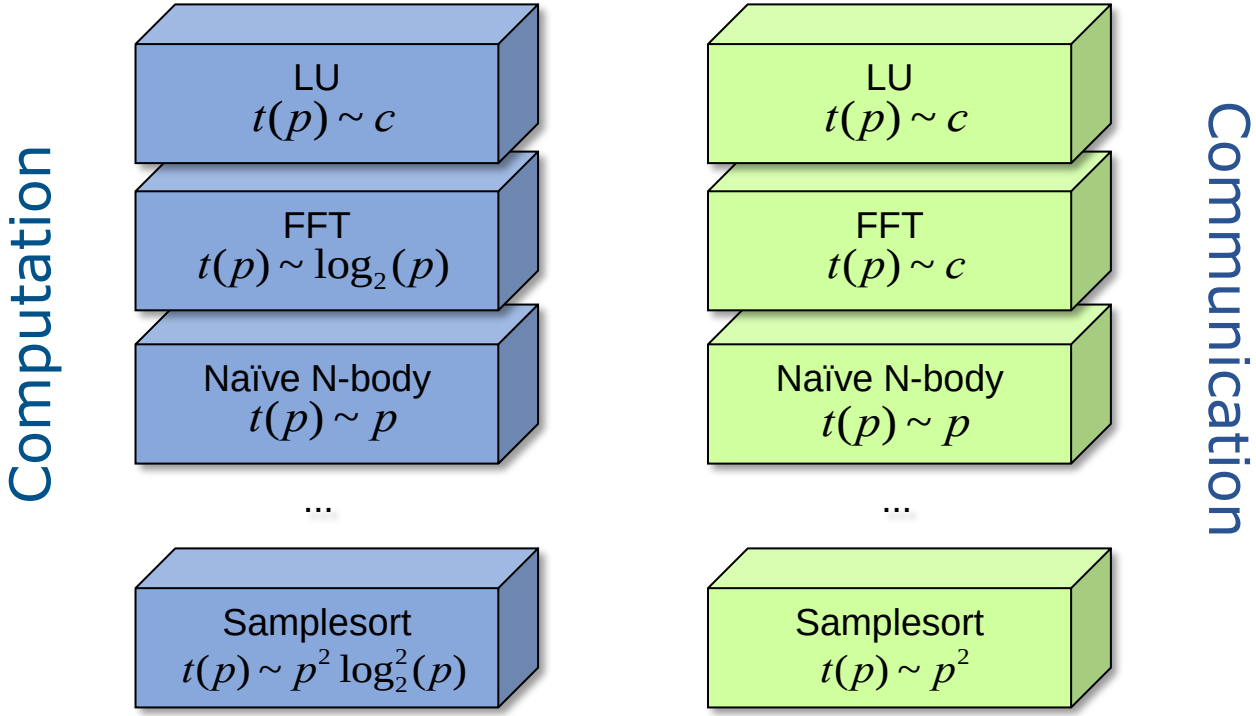
Primary focus on scaling trend



Ranking

1. F_2
2. F_1
3. F_3

Model building blocks

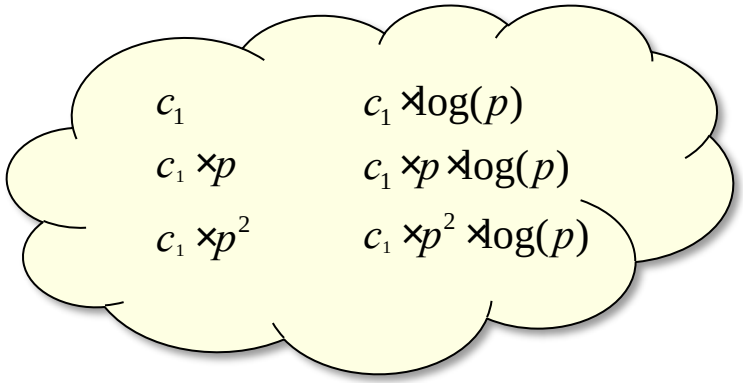


Performance model normal form

$$f(p) = \sum_{k=1}^{n} c_k \cdot p^{i_k} \cdot \log_2^{j_k}(p)$$

$n \in \mathbb{N}$
$i_k \in I$
$j_k \in J$
$I, J \subset \mathbb{Q}$

$n = 1$
$I = \{0, 1, 2\}$
$J = \{0, 1\}$



Performance model normal form

$$f(p) = \sum_{n} \sum_{i \in I} \sum_{j \in J} c_{n,i,j} \cdot p^{i} \cdot \log(p)^{j}$$

$n = 1$

$I = \{0, 1, 2\}$

$J = \{0, 1\}$

$c_1 + c_2 \cdot p$

$c_1 + c_2 \cdot p^2$

$c_1 + c_2 \cdot \log(p)$

$c_1 + c_2 \cdot p \cdot \log(p)$

$c_1 + c_2 \cdot p^2 \cdot \log(p)$

$c_1 \cdot \log(p) + c_2 \cdot p$

$c_1 \cdot \log(p) + c_2 \cdot p \cdot \log(p)$

$c_1 \cdot \log(p) + c_2 \cdot p^2$

$c_1 \cdot \log(p) + c_2 \cdot p^2 \cdot \log(p)$

$c_1 \cdot p + c_2 \cdot p \cdot \log(p)$

$c_1 \cdot p + c_2 \cdot p^2$

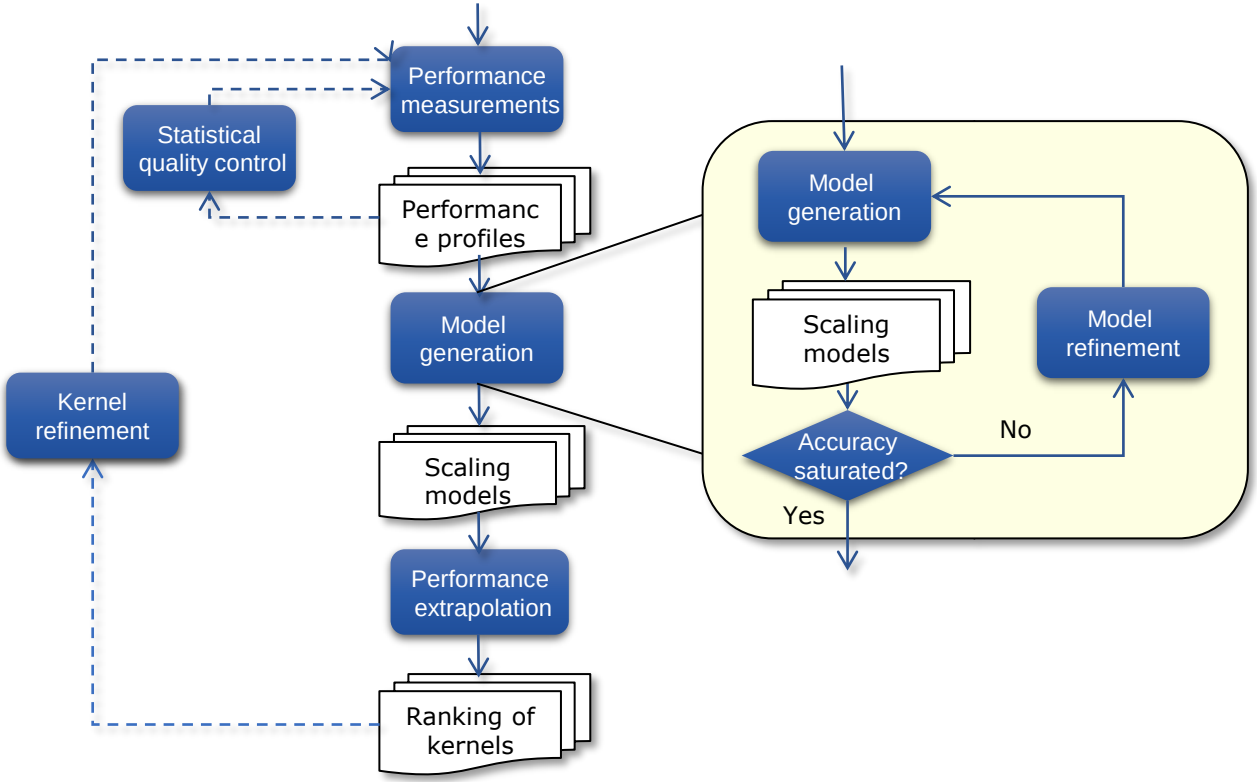
$c_1 \cdot p + c_2 \cdot p^2 \cdot \log(p)$

$c_1 \cdot p \cdot \log(p) + c_2 \cdot p^2$

$c_1 \cdot p \cdot \log(p) + c_2 \cdot p^2 \cdot \log(p)$

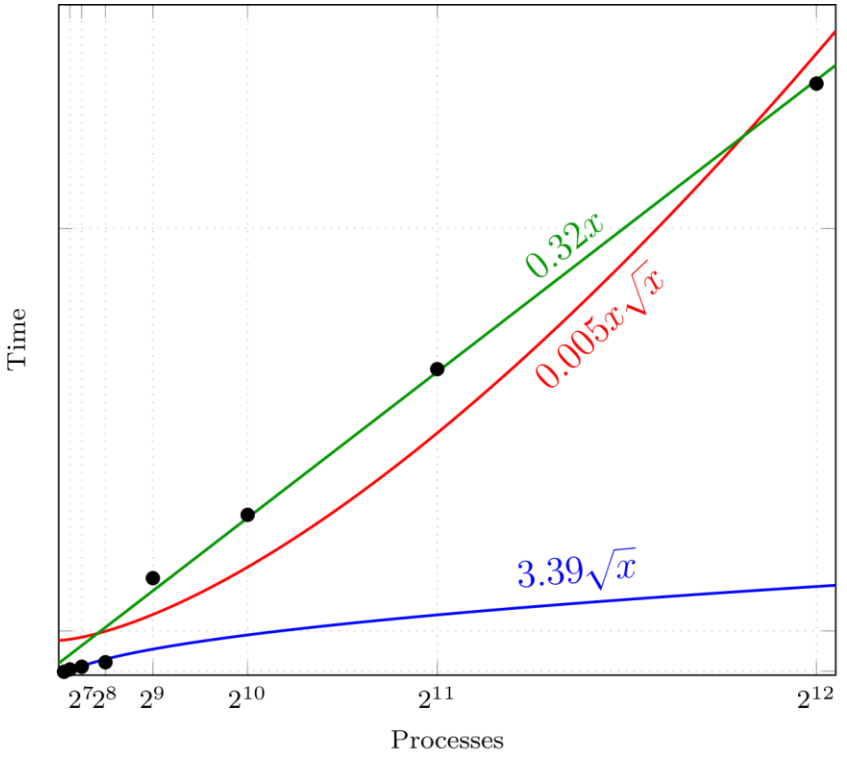
$c_1 \cdot p^2 + c_2 \cdot p^2 \cdot \log(p)$

Workflow


Performance measurements
Statistical quality control
Performanc e profiles
Model generation
Scaling models
Performance extrapolation
Ranking of kernels
Kernel refinement
Model generation
Scaling models
Accuracy saturated?
No
Yes
Model refinement

Assumptions & limitations

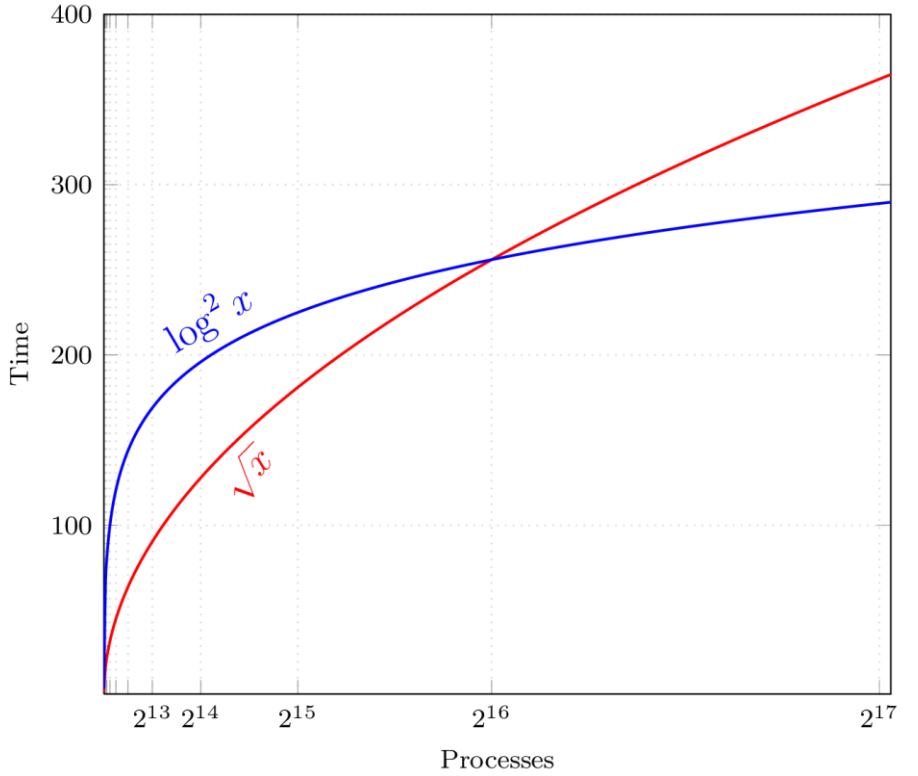
- Only one scaling behavior for all the measurements; no jumps
- Some MPI collective operations switch their algorithm – results in bad models
- Example: red model tries to model measurements of different algorithms
  - First 4 points – one function
  - Last 4 points – another function (linear)
  - Adj. R2 = 0.95085 (!)



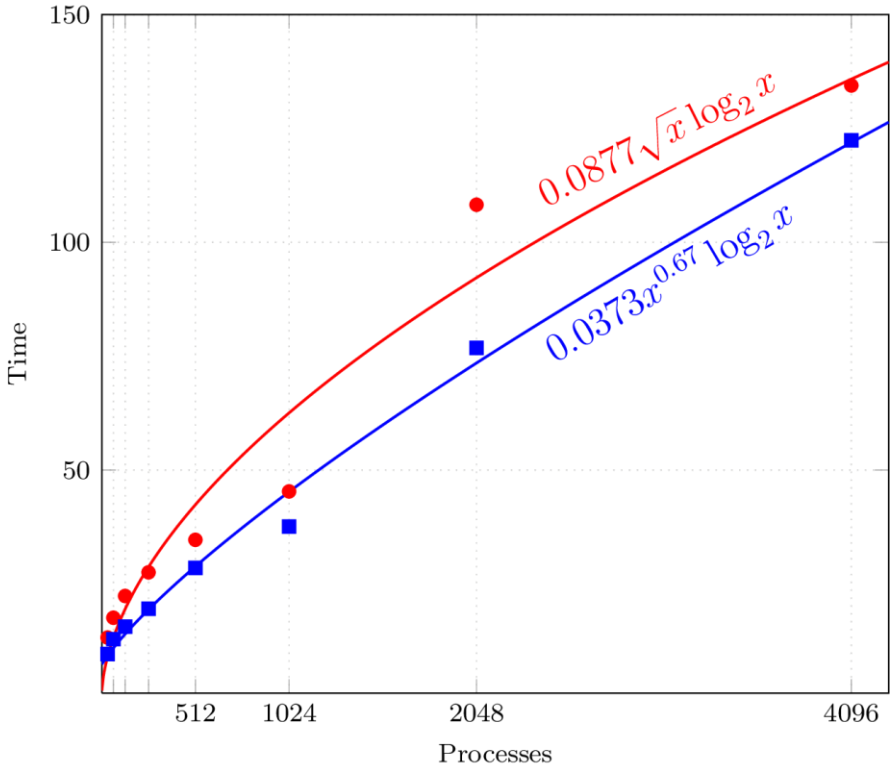
Changing growth trends

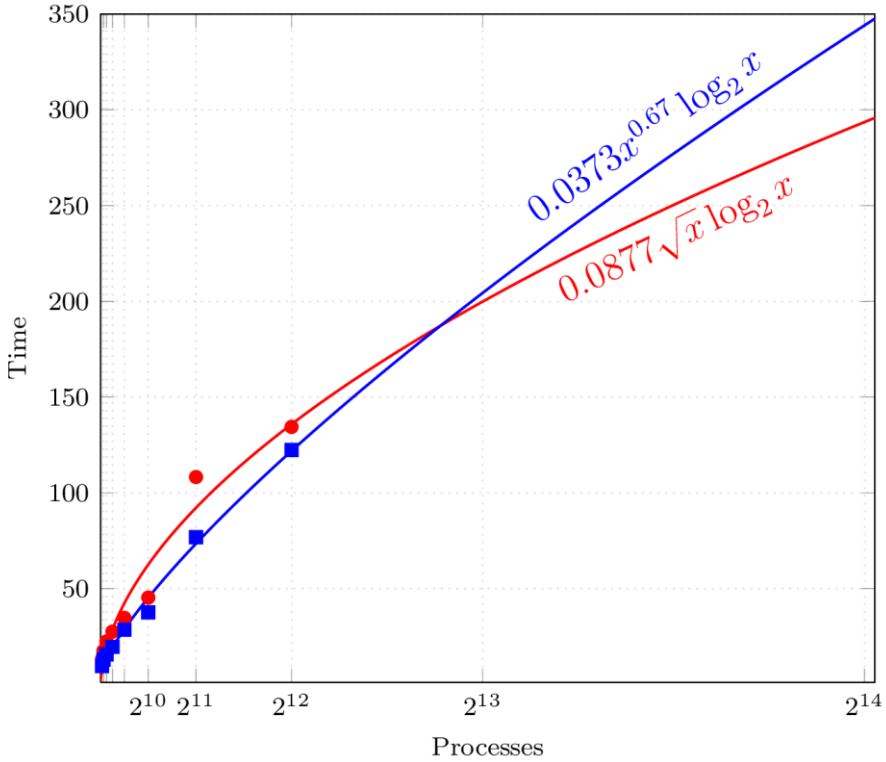
- Ranking according to growth rate difficult:

  $\log^2(p)$? $\sqrt{p}$



Changing growth trends (2)





Ranking of kernels

- Kernels are ranked according the leading-order terms in the models
- Leading-order term → big-O notation
- For example: $O(x)$ comes before $O(x^2)$

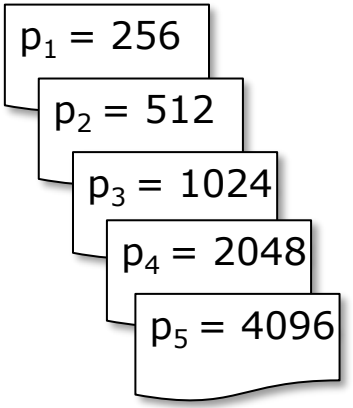
Performance measurements

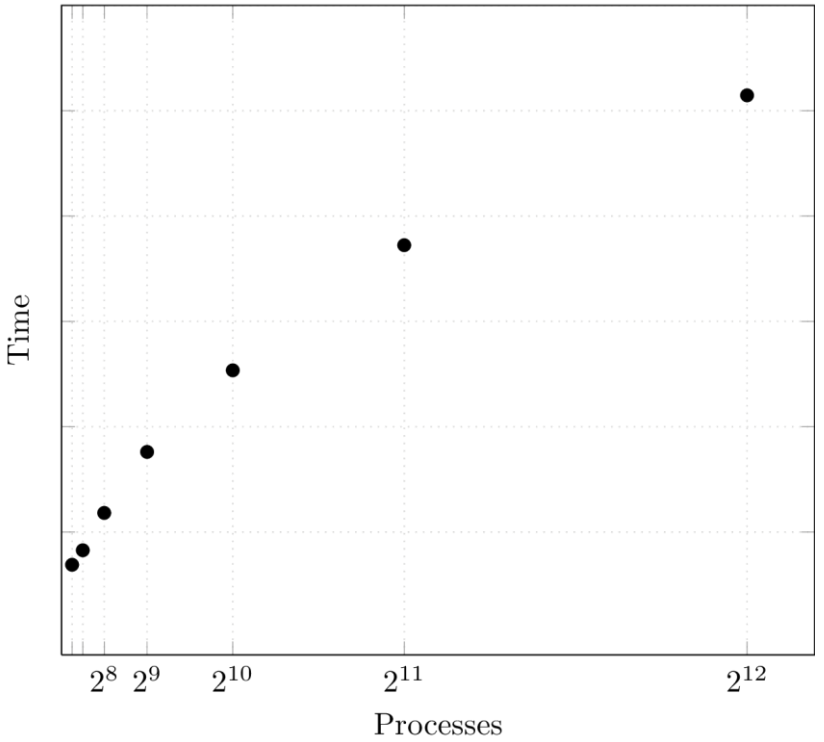
- Different ways of collecting measurements
- Score-P (http://www.vi-hps.org/projects/score-p/)
- Other profiling tools, e.g. HPCToolkit
- Manual ad-hoc measurements

Performance measurements (2)

- At least 5 different measurements required

Performance measurements (profiles)

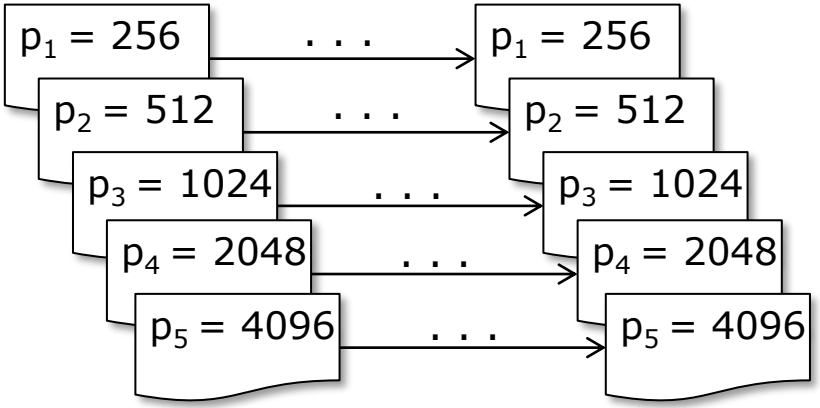


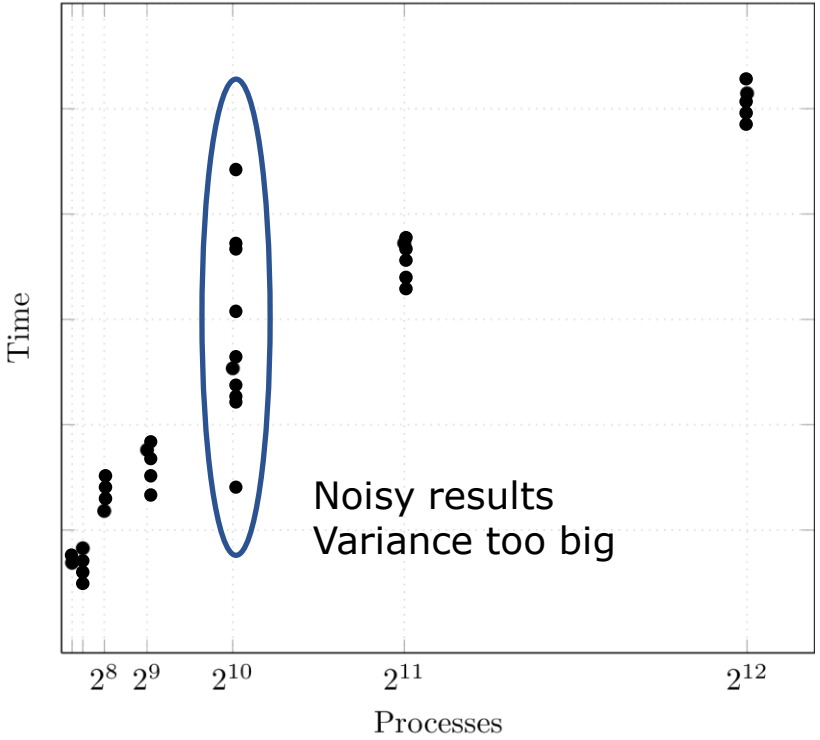


Performance measurements (3)

- At least 5 different measurements required
- Each measurement repeated multiple times

Performance measurements (profiles)



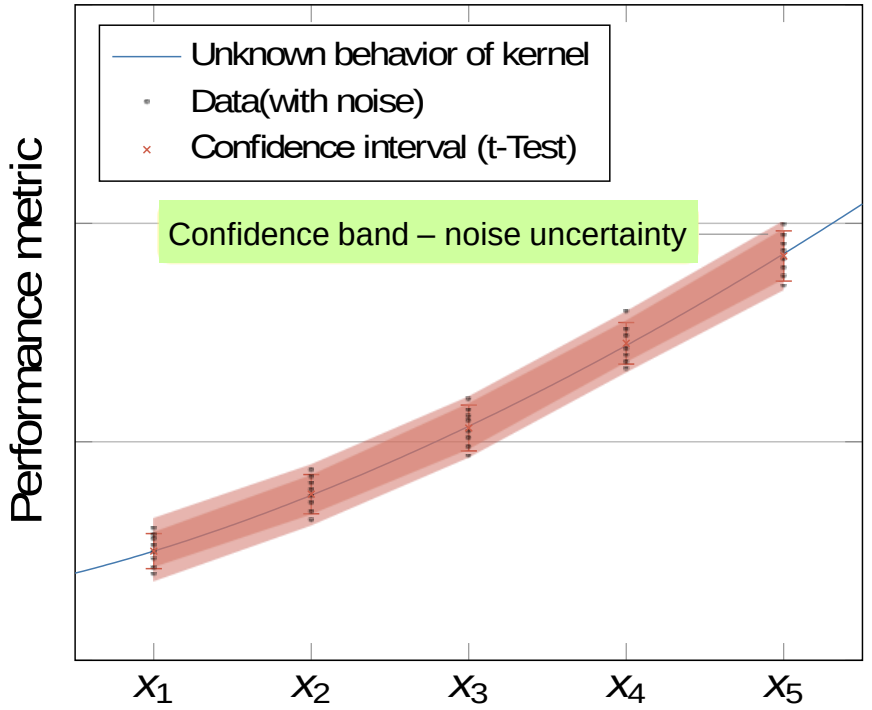


Statistical quality control

- If the confidence interval is too wide, the fit will not be optimal, or overfitting might occur

  CI = f(mean, stddev)

- To improve CI - increase repetitions, include different configurations



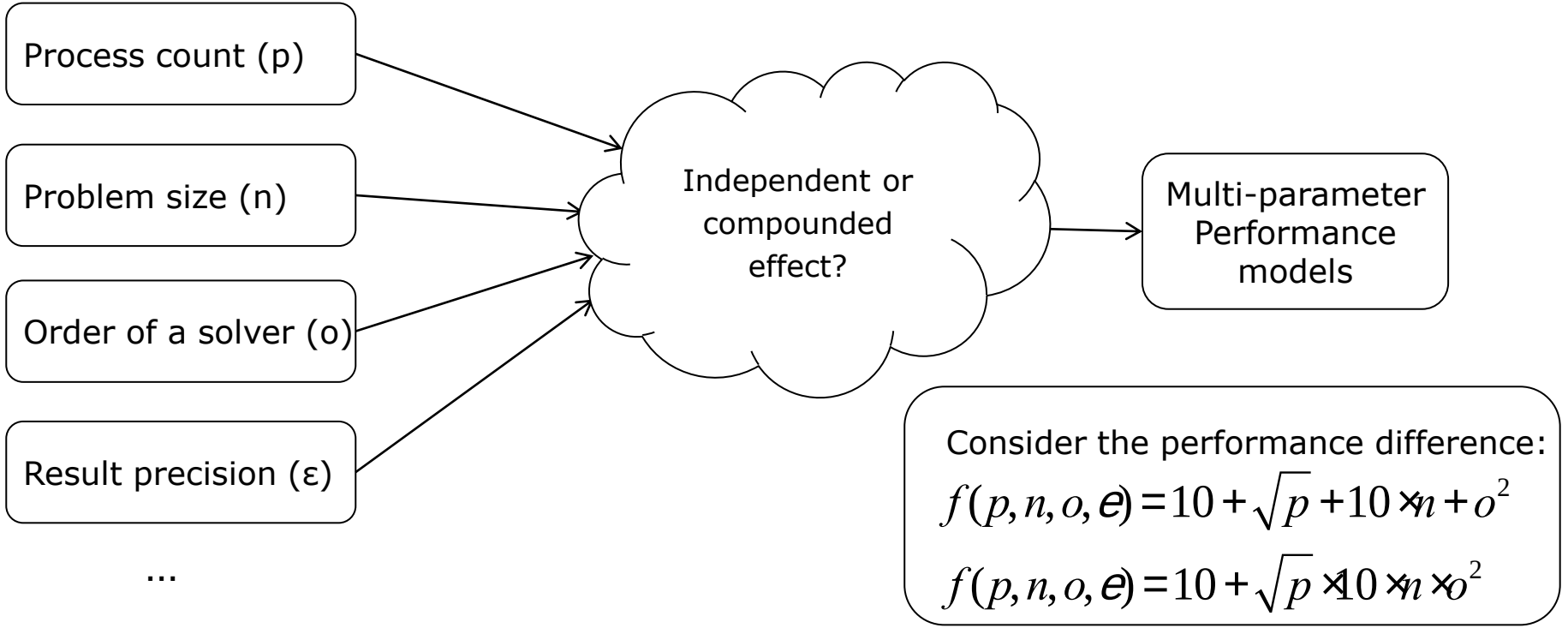
Adjusted R^2

- R^2 represents how well the determined function fits the M available measurements
- Adjusted R^2 adjusts for N, the number of terms used
  - Adj. R^2 decreases → more useless variables
  - Adj. R^2 increases → more useful variables
- Rule of thumb: adj. $R^2 > 0.95$

$$R^2 = 1 - \frac{residualSumSquares}{totalSumSquares}$$

$$\overline{R}^2 = 1 - (1 - R^2) \times \frac{M - 1}{M - N - 2}$$

Extra-P 3.0: Fast multi-parameter performance modeling



Extra-P 3.0: Fast multi-parameter performance modeling

Expanded performance model normal form

$$f(p) = \sum_{k=1}^{n} c_k \prod_{l=1}^{m} p_l^{i_{kl}} \times \log_2^{j_{kl}}(p_l)$$

$n \in \mathbb{N}$
$m \in \mathbb{N}$
$i_{kl} \in I$
$j_{kl} \in J$
$I, J \subset \mathbb{Q}$

$n = 2$
$m = 2$
$I = \left\{\frac{0}{4}, \frac{1}{4}, \ldots, \frac{12}{4}\right\}$
$J = \{0, 1, 2\}$

Model candidates:

- Constant c_1
- Single parameter $c_1 + c_2 \times p_1$
- Multiple parameters …
  - Additive $c_1 + c_2 \times p_1 + c_3 \times p_2$
  - Multiplicative $c_1 + c_2 \times p_1 \times p_2$
  - Complex $c_1 + c_2 \times p_1 + c_3 \times p_1^2 \times p_2 \times \log_2(p_2)$

Extra-P 3.0: Fast multi-parameter performance modeling

- **Hierarchical search** – Reduces search from all possible models to combinations of the best single parameter models of each parameter
- **Modified golden section search** – Orders single parameter search space and applies a modified binary search
- Assuming 300.000 models searched per second* – 3 parameter models

Exhaustive search	$C(59.319, 3) = 34.786.300.841.019$	~3.5 years/model
Hierarchical search	$3 \times 9.139 + 512 = 27.929$	~11 models/second
Hierarchical search + Modified golden section search	$3 \times 26 + 512 = 590$	~508 models/second

*This is a simplification, multi-parameter models take much longer to be evaluated

Using Extra-P 2.0

Extra-P in the tuning workshop

- Installed in: `/home/hpc/a2c06/lu23voj/…`
- Run: `source /home/hpc/a2c06/lu23voj/load-extrap`
- Extra-P executables: `extrap`, `extrap-print`

Installing Extra-P

- Download and install Qt4 or Qt5
- Download and install Python3+ and PyQt5 (Extra-P 3.0 requires matplotlib)
- Download Extra-P: http://www.scalasca.org/software/extra-p/download.html
- Unpack & install Extra-P
- `./configure --prefix=<extra-p-install-dir> CPPFLAGS=-I<python.h path> PYTHON=<Python interpreter>; make; make install`

Automatic performance modeling with Extra-P

Input

```
Myapp.cpp
main() {
  foo()
  bar()
  compute()
}
```

Instrumentation & Measurement

Score-P

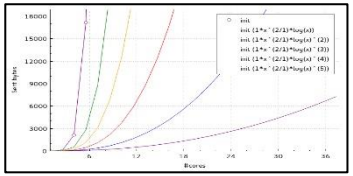
Performance measurements (profiles)

- Myapp.weak.p128.r1/profile.cubex
- Myapp.weak.p256.r1/profile.cubex
- Myapp.weak.p512.r1/profile.cubex
- Myapp.weak.p1024.r1/profile.cubex
- Myapp.weak.p2048.r1/profile.cubex

Extra-P

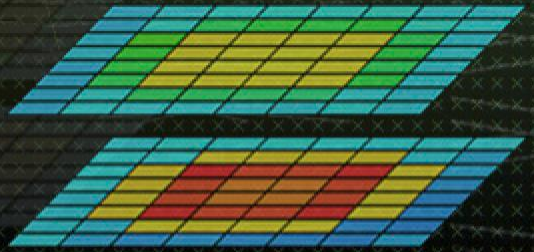
Output

Results (visual)



Results (textual)

```
Region 1: main
Model: (3) + (3.14 * x^( 2 ))
...
```



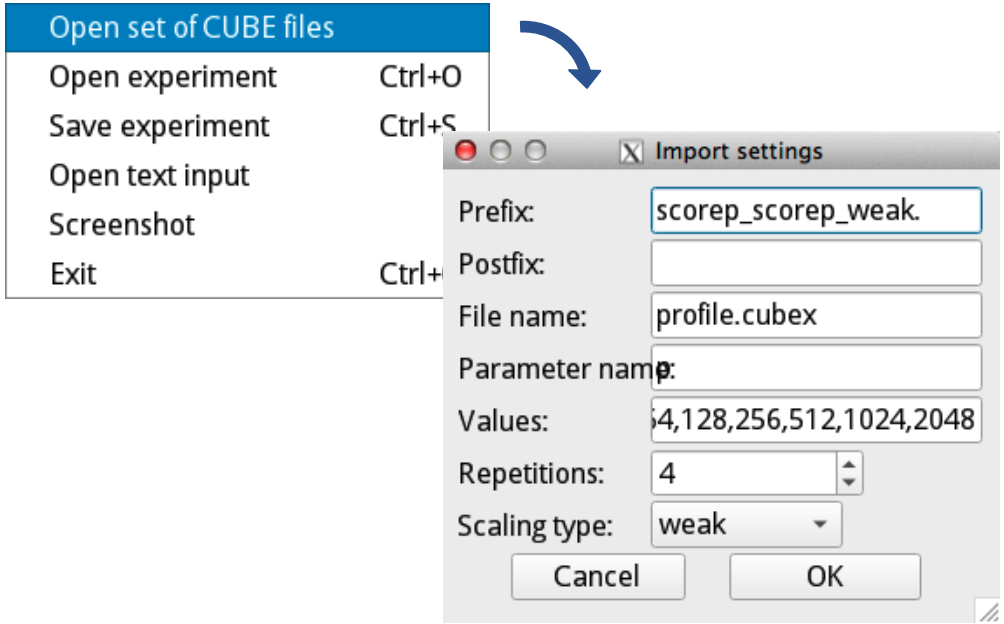
Modeling sets of Cube experiments

Extra-P Cube input description

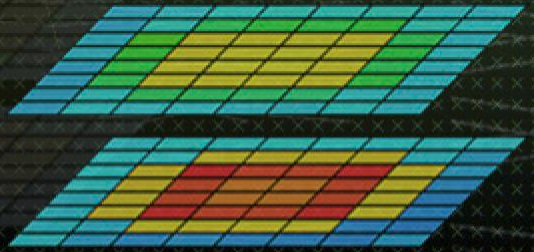
- Modeling tool expects Cube files in the following format:
  <DIR>/<PREFIX><X><POSTFIX>.r<{1,..,REPS}>/<FILENAME>
- DIR, PREFIX, X, POSTFIX, REPS and FILENAME must all be defined.
  - X – value of varied parameter e.g. number of processes
  - REPS – number of repeated experiments with same parameter value

Extra-P Cube input description

<DIR>/<PREFIX><X><POSTFIX>.r<{1,..,REPS}>/<FILENAME>

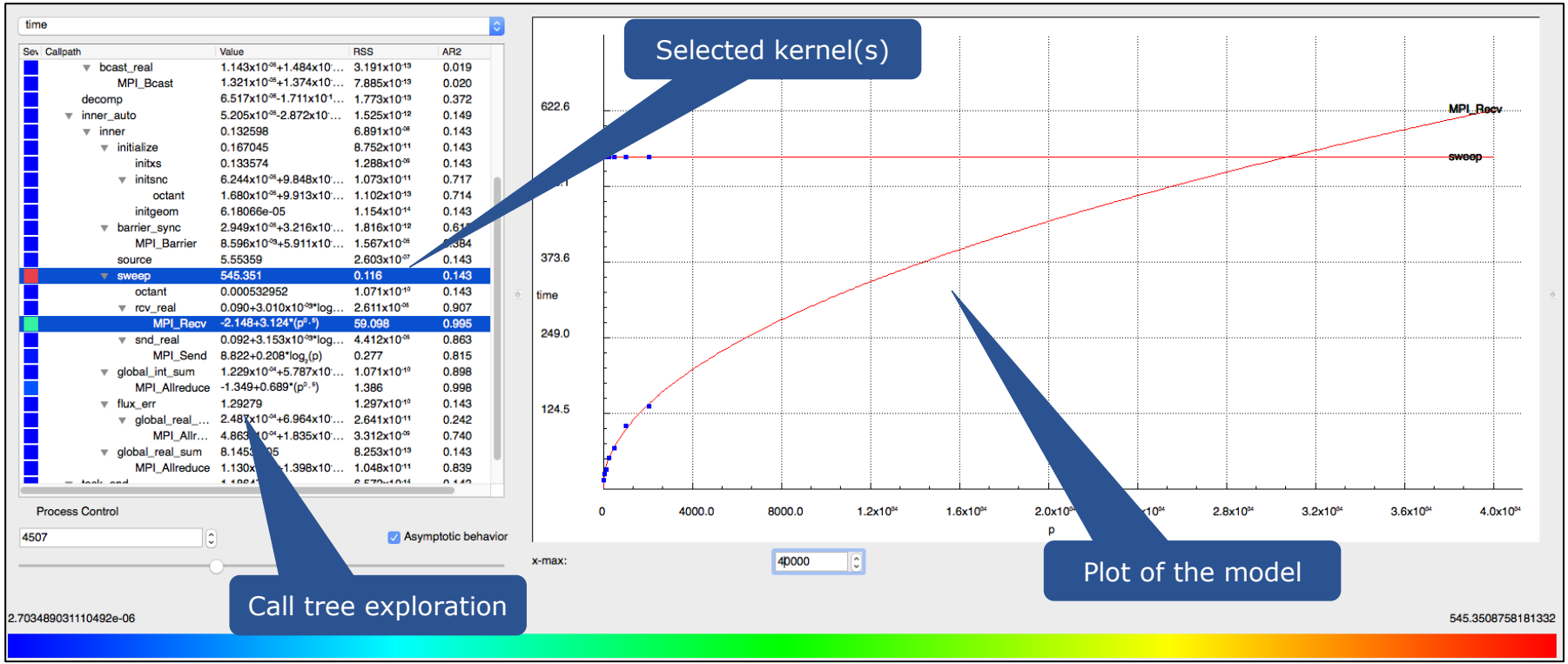


Best effort approach to identify and populate the fields automatically based on selected directory contents.



Visualization with Extra-P

Extra-P user interface



Extra-P call tree view

Metric selection

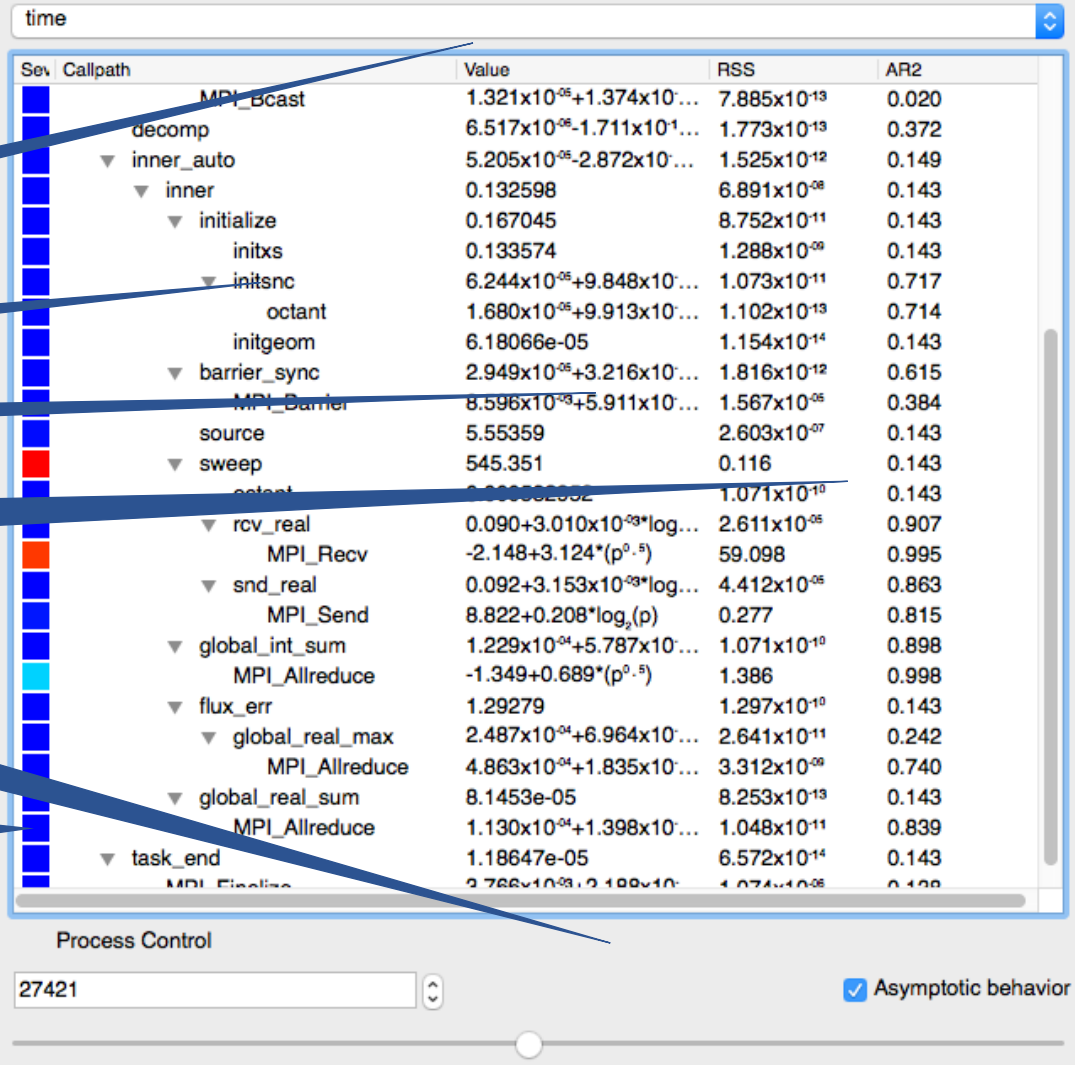
Call tree exploration

Model

Quality of fit metrics: Residual sum of squares and Adjusted R^2

Asymptotic view of model functions vs. value at given value

Impact of each kernel on the metric at the selected process count compared to the other kernels

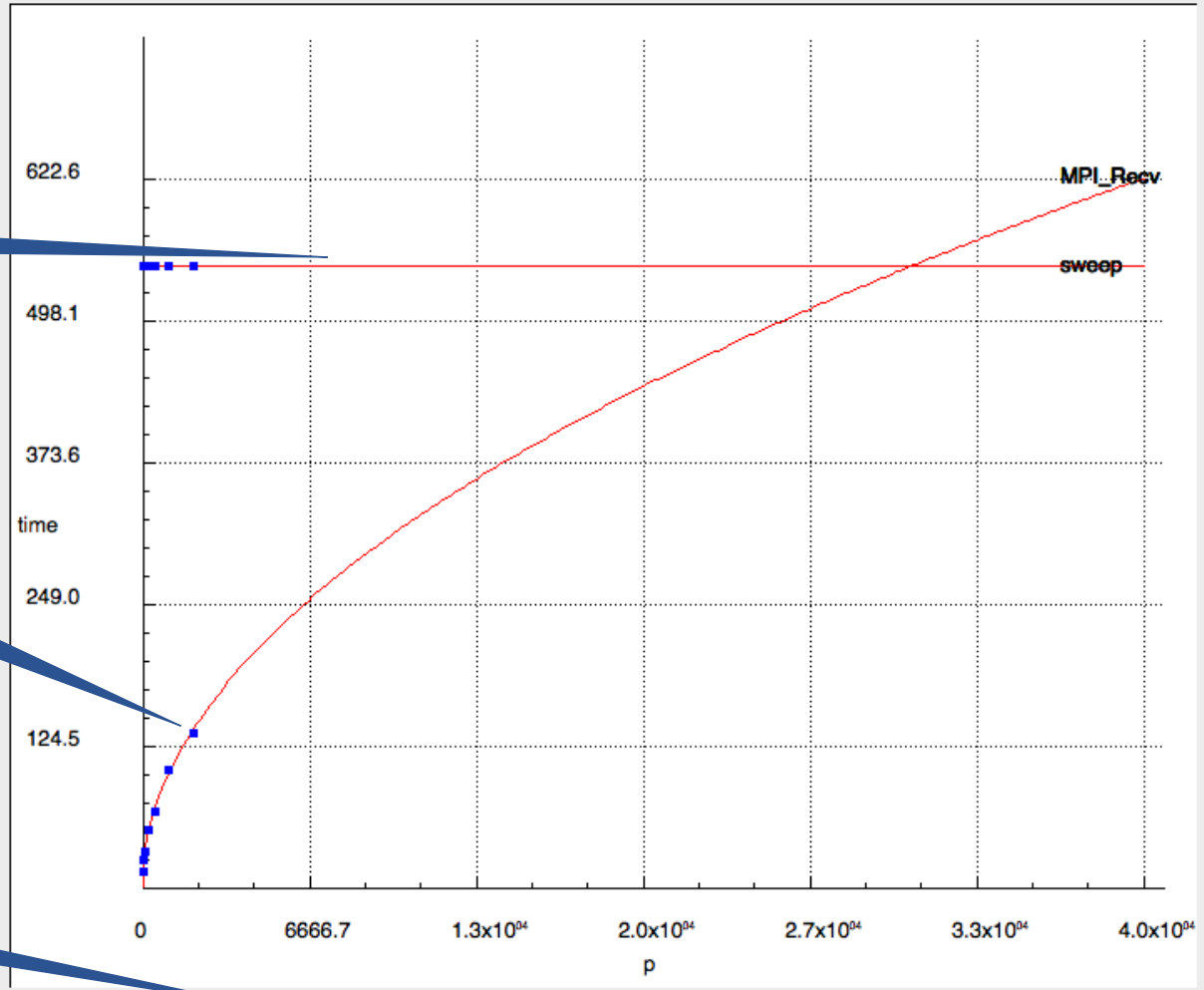


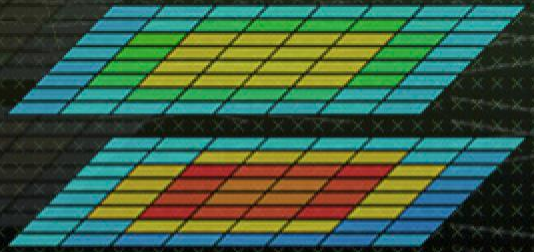
Extra-P model view

Models selected in the Call path view

Measurement values

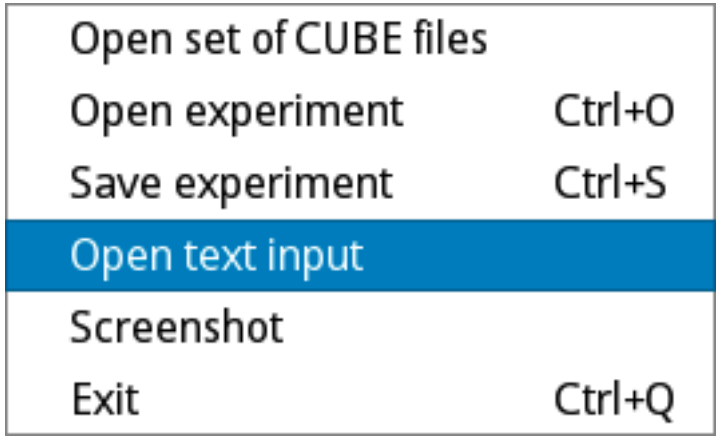
X axis scale control for prediction of behavior at other process counts





Modeling measurements from a text file

Choose input file



Select input file in the GUI

Extra-P input in text form

- Useful when no CUBE files are available or when a small data set must be modeled
- Example provided in `/home/hpc/a2c06/lu23voj/tutorial/experiment2.txt`

```
PARAMETER p
POINTS 1000 2000 4000 8000 16000
METRIC metric1
REGION region1
DATA 1 1 1 1 1
DATA 4 4 4 3.99 4.01
DATA 16 15.999 16.01 16.01 15.99
DATA 64 64 64 64.01 63.99
DATA 256.01 255.99 256 256
```

Parameter name
This name will be used in the GUI as well as in the textual output

Extra-P input in text form

- Useful when no CUBE files are available or when a small data set must be modeled
- Example provided in `/home/hpc/a2c06/lu23voj/tutorial/experiment2.txt`

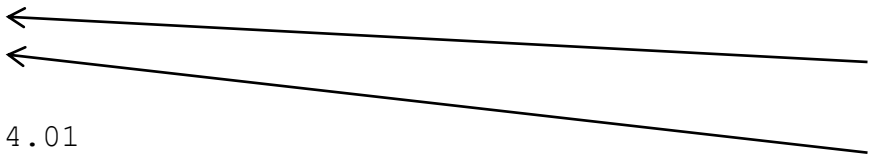
```
PARAMETER p
POINTS 1000 2000 4000 8000 16000
METRIC metric1
REGION region1
DATA 1 1 1 1 1
DATA 4 4 4 3.99 4.01
DATA 16 15.999 16.01 16.01 15.99
DATA 64 64 64 64.01 63.99
DATA 256.01 255.99 256 256
```

Measurement points
Use at least 5, preferably 6,
but in general the more the better

Extra-P input in text form

- Useful when no CUBE files are available or when a small data set must be modeled
- Example provided in `/home/hpc/a2c06/lu23voj/tutorial/experiment2.txt`

```
PARAMETER p
POINTS 1000 2000 4000 8000 16000
METRIC metric1
REGION region1
DATA 1 1 1 1 1
DATA 4 4 4 3.99 4.01
DATA 16 15.999 16.01 16.01 15.99
DATA 64 64 64 64.01 63.99
DATA 256.01 255.99 256 256
```



Metric name

Region name

Both used to determine the output Cube file hierarchical structure and identify separate data sets

Extra-P input in text form

- Useful when no CUBE files are available or when a small data set must be modeled
- Example provided in `/home/hpc/a2c06/lu23voj/tutorial/experiment2.txt`

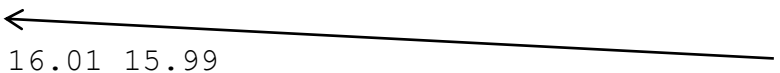
```
PARAMETER p
POINTS 1000 2000 4000 8000 16000
METRIC metric1
REGION region1
DATA 1 1 1 1 1
DATA 4 4 4 3.99 4.01
DATA 16 15.999 16.01 16.01 15.99
DATA 64 64 64 64.01 63.99
DATA 256.01 255.99 256 256
```

Data points
Each row corresponds to a point; all values in a row are considered repeat measurements of the same experiment

Extra-P input in text form

- Useful when no CUBE files are available or when a small data set must be modeled
- Example provided in `/home/hpc/a2c06/lu23voj/tutorial/experiment2.txt`

```
PARAMETER p
POINTS 1000 2000 4000 8000 16000
METRIC metric1
REGION region1
DATA 1 1 1 1 1
DATA 4 4 4 3.99 4.01
DATA 16 15.999 16.01 16.01 15.99
DATA 64 64 64 64.01 63.99
DATA 256.01 255.99 256 256
```



Data points
Each row corresponds to a point; all values in a row are considered repeat measurements of the same experiment

Extra-P input in text form

- Useful when no CUBE files are available or when a small data set must be modeled
- Example provided in `/home/hpc/a2c06/lu23voj/tutorial/experiment2.txt`

```
PARAMETER p
POINTS 1000 2000 4000 8000 16000
METRIC metric1
REGION region1
DATA 1 1 1 1 1
DATA 4 4 4 3.99 4.01
DATA 16 15.999 16.01 16.01 15.99
DATA 64 64 64 64.01 63.99
DATA 256.01 255.99 256 256
```

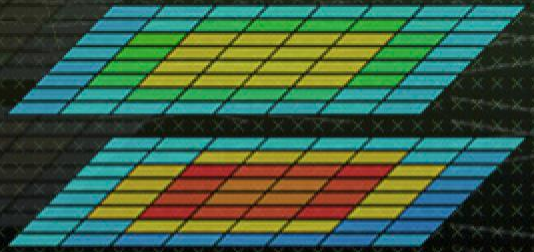
Data points
Each row corresponds to a point; all values in a row are considered repeat measurements of the same experiment

Extra-P input in text form

- Useful when no CUBE files are available or when a small data set must be modeled
- Example provided in `/home/hpc/a2c06/lu23voj/tutorial/experiment2.txt`

```
PARAMETER p
POINTS 1000 2000 4000 8000 16000
METRIC metric1
REGION region1
DATA 1 1 1 1 1
DATA 4 4 4 3.99 4.01
DATA 16 15.999 16.01 16.01 15.99
DATA 64 64 64 64.01 63.99
DATA 256.01 255.99 256 256
```

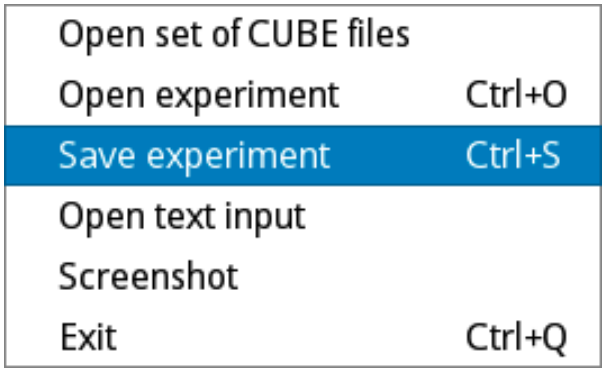
Data points
Each row corresponds to a point; all values in a row are considered repeat measurements of the same experiment



Results in text form

Produce text output

- Step 1: Save results

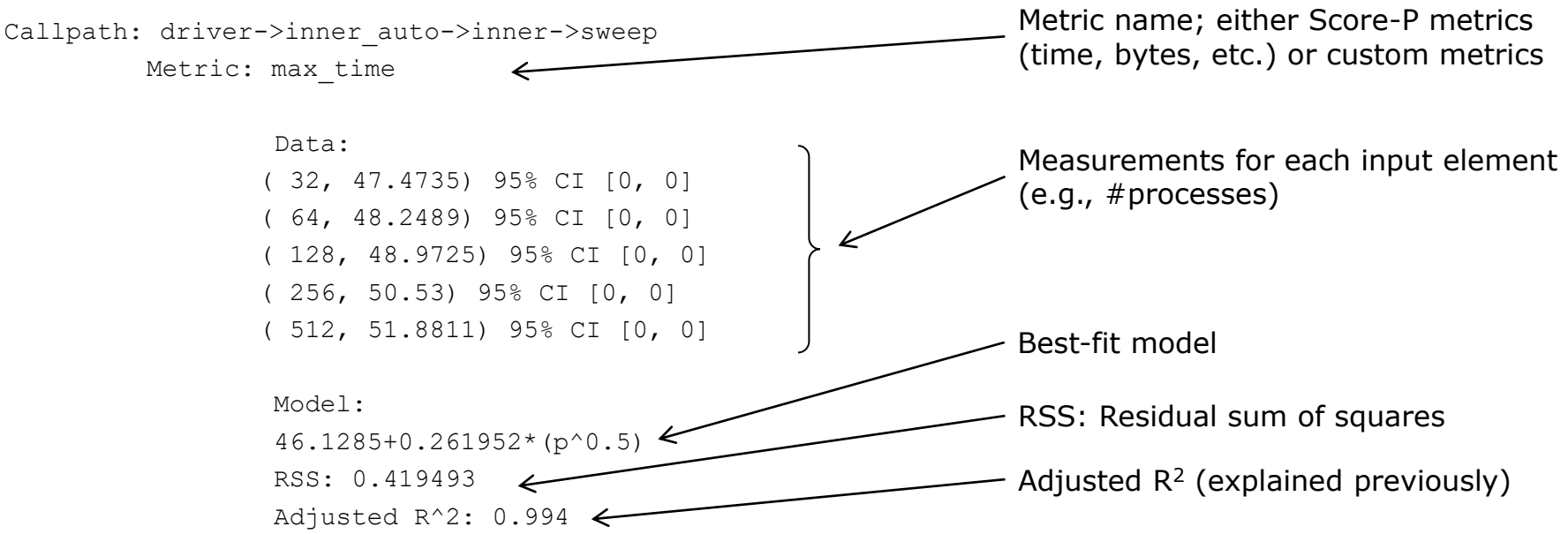


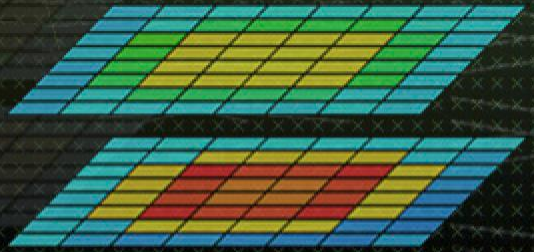
Save the experiment currently open in Extra-P to an output file

- Step 2: Run `extrap-print <Extra-P experiment file>`

Text output format

Measurements and model data for each experiment and metric:





Hands-on exercises

Extra-P exercises

- Run: `extrap`
- Examples: `/home/hpc/a2c06/lu23voj/tutorial/{blast,sweep3D}`
- Open the examples in the GUI
- Open the textual input example
- Produce textual output and inspect it

Feedback

- What additional features would you like to see?
- What additional capabilities would you like to see?