Optimization using modelization and experiment design

Steven QUINITO MASNADA

April 4, 2016

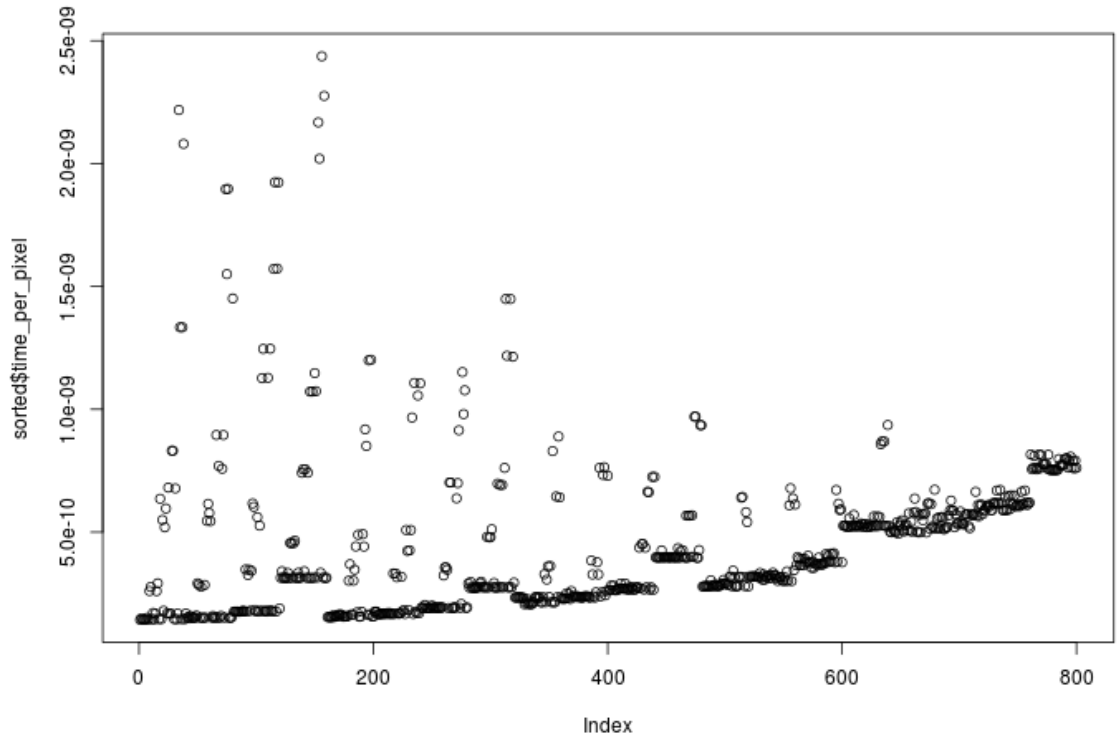
Topic

Laplacian

- Parameters:
  - x_component_number [1,2,4,8,16]
  - y_component_number [1,2,3,4]
  - vector_length [1,2,4,8,16]
  - temporary_size [2,4]
  - vector_recompute [true,false]
  - load_overlap [true,false]
- 800 combinations
- OpenCL Nvidia implementation

A structured search space

- Found what are the parameters that have an impact
- Hierarchy of parameters:
  - x_component_number → y_component_number → vector_recompute → vector_length



A simple approach : Linear model

- Global optimum :

x_component_number	1	y_component_number	4
load_overlap	false	vector_length	1
vector_recompute	false	temporary_size	4

- A simple model allowed us to find the structure of the search space

```
Call:
lm(formula = time_per_pixel ~ x_component_number + vector_length +
y_component_number + vector_recompute, data = df)

Residuals:
  Min          1Q      Median          3Q         Max
-3.494e-10 -1.409e-10 -3.041e-11  8.200e-11  1.627e-09

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)           2.770e-10  2.622e-11  10.567   <2e-16 ***
x_component_number    1.364e-11  1.552e-12   8.785   <2e-16 ***
vector_length         2.308e-11  1.552e-12  14.870   <2e-16 ***
y_component_number   -6.680e-11  7.575e-12  -8.819   <2e-16 ***
vector_recomputetrue  2.162e-10  1.694e-11  12.763   <2e-16 ***
---
```

Topic

Tuning the work size

- Local work size : [32,1], [32,2], [32,4], [32,8], [64,1] . . . , [256,1]
- Vector_recompute and load_overlap : true
- Max 256 threads per work groups
- 2000 combinations
- Global optimum:
  - x_component_number: 1
  - y_component_number: 4
  - vector_length: 1
  - temporary_size: 4
  - load_overlap: true
  - vector_recompute: true
  - local_work_size [128,2]

Finding revelant parameters

```
Call:
lm(formula = time_per_pixel ~ x_component_number + y_component_number +
    vector_length + temporary_size + factor(local_work_size),
    data = df)

Residuals:
       Min          1Q      Median          3Q         Max
-4.124e-10 -1.586e-10 -5.260e-11  1.101e-10  1.413e-09

Coefficients:
                                    Estimate Std. Error t value Pr(>|t|)
(Intercept)                         4.817e-10  2.810e-11  17.145  < 2e-16 ***
x_component_number                 -1.027e-11  9.989e-13 -10.283  < 2e-16 ***
y_component_number                 -8.007e-11  4.874e-12 -16.428  < 2e-16 ***
vector_length                       3.376e-11  9.989e-13  33.801  < 2e-16 ***
temporary_size                     -8.882e-14  5.449e-12  -0.016  0.98700
factor(local_work_size)[128, 2, 1]  1.118e-11  2.437e-11   0.459  0.64637
factor(local_work_size)[256, 1, 1]  1.568e-11  2.437e-11   0.644  0.51993
factor(local_work_size)[32, 1, 1]   7.581e-11  2.437e-11   3.111  0.00189 **
factor(local_work_size)[32, 2, 1]   2.476e-12  2.437e-11   0.102  0.91907
factor(local_work_size)[32, 4, 1]  -2.783e-12  2.437e-11  -0.114  0.90910
factor(local_work_size)[32, 8, 1]   7.394e-12  2.437e-11   0.303  0.76161
factor(local_work_size)[64, 1, 1]   2.301e-11  2.437e-11   0.944  0.34508
factor(local_work_size)[64, 2, 1]  -1.268e-12  2.437e-11  -0.052  0.95851
```

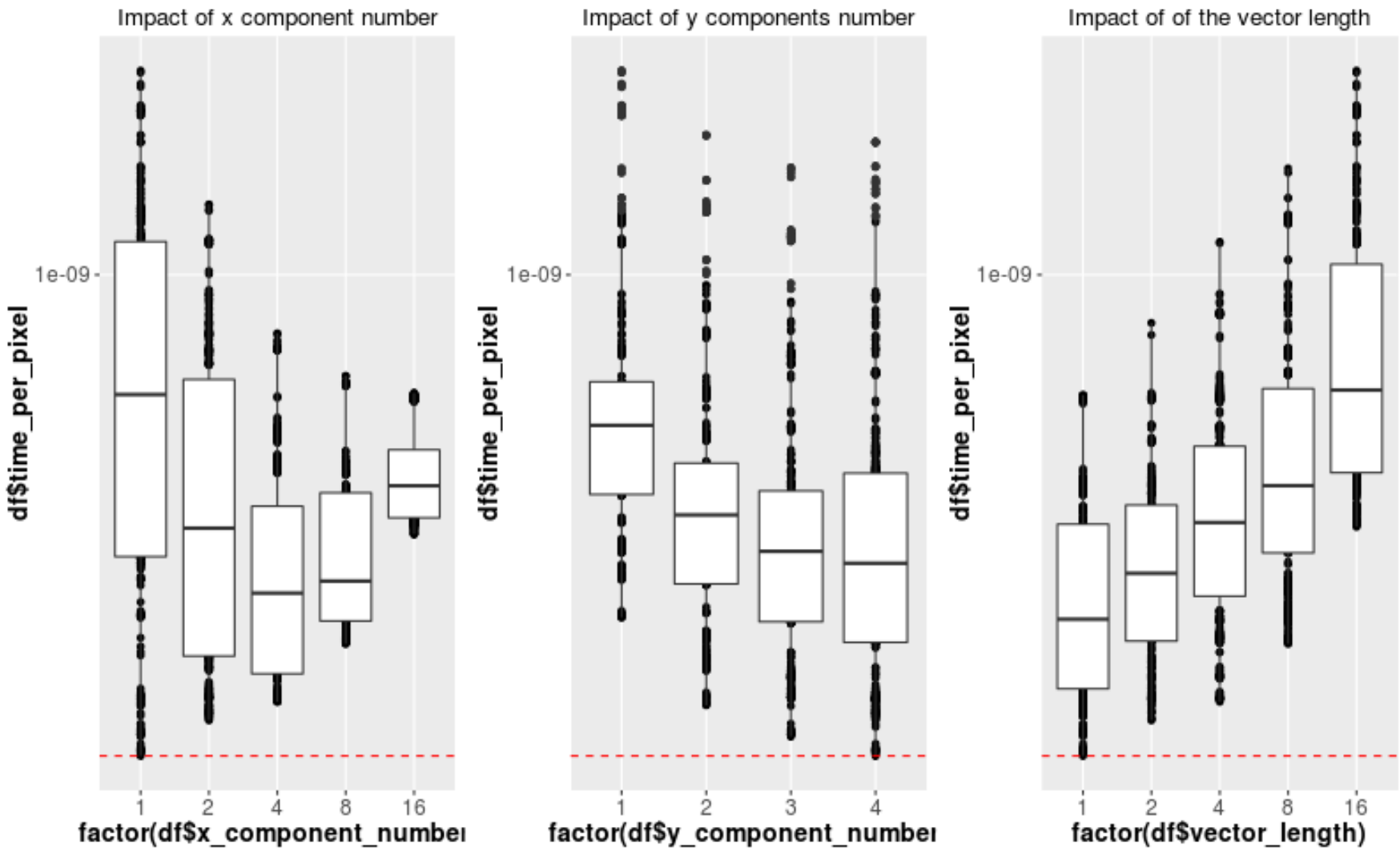
Finding revelant parameters

```
Call:
lm(formula = time_per_pixel ~ x_component_number + y_component_number +
    vector_length + temporary_size + factor(local_work_size),
    data = df)

Residuals:
       Min          1Q      Median          3Q         Max
-4.124e-10 -1.586e-10 -5.260e-11  1.101e-10  1.413e-09
```

R-squared = 0.4359

```
                                    4.817e-10  2.810e-11  17.145  < 2e-16 ***
x_component_number                 -1.027e-11  9.989e-13 -10.283  < 2e-16 ***
y_component_number                 -8.007e-11  4.874e-12 -16.428  < 2e-16 ***
vector_length                       3.376e-11  9.989e-13  33.801  < 2e-16 ***
temporary_size                     -8.882e-14  5.449e-12  -0.016  0.98700
factor(local_work_size)[128, 2, 1]  1.118e-11  2.437e-11   0.459  0.64637
factor(local_work_size)[256, 1, 1]  1.568e-11  2.437e-11   0.644  0.51993
factor(local_work_size)[32, 1, 1]   7.581e-11  2.437e-11   3.111  0.00189 **
factor(local_work_size)[32, 2, 1]   2.476e-12  2.437e-11   0.102  0.91907
factor(local_work_size)[32, 4, 1]  -2.783e-12  2.437e-11  -0.114  0.90910
factor(local_work_size)[32, 8, 1]   7.394e-12  2.437e-11   0.303  0.76161
factor(local_work_size)[64, 1, 1]   2.301e-11  2.437e-11   0.944  0.34508
factor(local_work_size)[64, 2, 1]  -1.268e-12  2.437e-11  -0.052  0.95851
```

Finding revelant parameters



Finding interactions

```
Call:
lm(formula = time_per_pixel ~ x_component_number * y_component_number *
    vector_length, data = df)

Residuals:
       Min          1Q      Median          3Q         Max
-4.722e-10 -1.029e-10 -1.970e-11  9.081e-11  1.130e-09

Coefficients:
                                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                              2.452e-10  2.520e-11   9.730  < 2e-16 ***
x_component                              2.071e-11  3.051e-12   6.788 1.49e-11 ***
y_component                             -4.981e-11  9.201e-12  -5.414 6.93e-08 ***
vector_length                            7.962e-11  3.051e-12  26.095  < 2e-16 ***
x_component:y_component                 -1.123e-12  1.114e-12  -1.008    0.314
x_component:vector_length               -5.881e-12  3.695e-13 -15.918  < 2e-16 ***
y_component:vector_length               -7.072e-12  1.114e-12  -6.347 2.71e-10 ***
x_component:y_component:vector_length    5.345e-13  1.349e-13   3.962 7.70e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.008e-10 on 1992 degrees of freedom
Multiple R-squared:  0.616,Adjusted R-squared:  0.6147
F-statistic: 456.5 on 7 and 1992 DF,  p-value: < 2.2e-16
```

Finding local actions

```
Call:
lm(formula = time_per_pixel ~ temporary_size + factor(local_work_size),
    data = df[df$x_component_number == 1 & df$y_component_number ==
        4 & df$vector_length == 1, ])

Residuals:
       Min          1Q      Median          3Q         Max
-3.576e-12 -1.469e-12  0.000e+00  1.469e-12  3.576e-12

Coefficients:
                                    Estimate Std. Error t value Pr(>|t|)
(Intercept)                         1.334e-10  2.612e-12  51.056 2.13e-12 ***
temporary_size                     -1.890e-12  5.993e-13  -3.154 0.011671 *
factor(local_work_size)[128, 2, 1]  3.680e-13  2.680e-12   0.137 0.893822
factor(local_work_size)[256, 1, 1] -8.104e-13  2.680e-12  -0.302 0.769234
factor(local_work_size)[32, 1, 1]   1.096e-10  2.680e-12  40.903 1.55e-11 ***
factor(local_work_size)[32, 2, 1]   1.896e-11  2.680e-12   7.073 5.84e-05 ***
factor(local_work_size)[32, 4, 1]   1.484e-12  2.680e-12   0.554 0.593215
factor(local_work_size)[32, 8, 1]  -3.554e-13  2.680e-12  -0.133 0.897422
factor(local_work_size)[64, 1, 1]   1.716e-11  2.680e-12   6.403 0.000125 ***
factor(local_work_size)[64, 2, 1]  -1.290e-12  2.680e-12  -0.481 0.641876
factor(local_work_size)[64, 4, 1]   6.897e-13  2.680e-12   0.257 0.802697
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

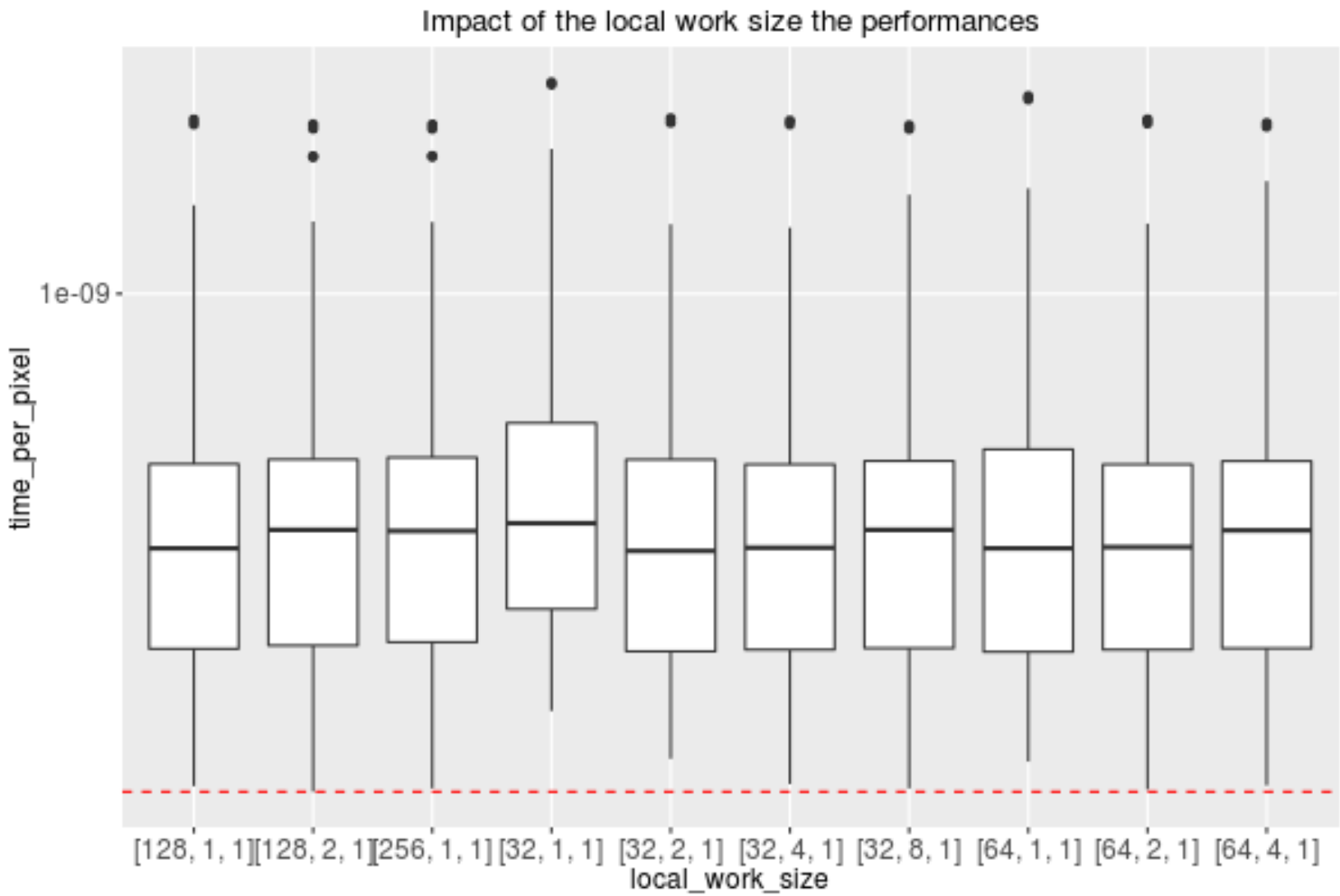
Finding local actions

```
Call:
lm(formula = time_per_pixel ~ temporary_size + factor(local_work_size),
    data = df[df$x_component_number == 1 & df$y_component_number ==
        4 & df$vector_length == 1, ])

Residuals:
```

Problem: the search space was not correctly expressed

```
Coefficients:
                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                      1.334e-10  2.612e-12  51.056 2.13e-12 ***
temporary_size                  -1.890e-12  5.993e-13  -3.154 0.011671 *
factor(local_work_size)[128, 2, 1]  3.680e-13  2.680e-12   0.137 0.893822
factor(local_work_size)[256, 1, 1] -8.104e-13  2.680e-12  -0.302 0.769234
factor(local_work_size)[32, 1, 1]   1.096e-10  2.680e-12  40.903 1.55e-11 ***
factor(local_work_size)[32, 2, 1]   1.896e-11  2.680e-12   7.073 5.84e-05 ***
factor(local_work_size)[32, 4, 1]   1.484e-12  2.680e-12   0.554 0.593215
factor(local_work_size)[32, 8, 1]  -3.554e-13  2.680e-12  -0.133 0.897422
factor(local_work_size)[64, 1, 1]   1.716e-11  2.680e-12   6.403 0.000125 ***
factor(local_work_size)[64, 2, 1]  -1.290e-12  2.680e-12  -0.481 0.641876
factor(local_work_size)[64, 4, 1]   6.897e-13  2.680e-12   0.257 0.802697
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

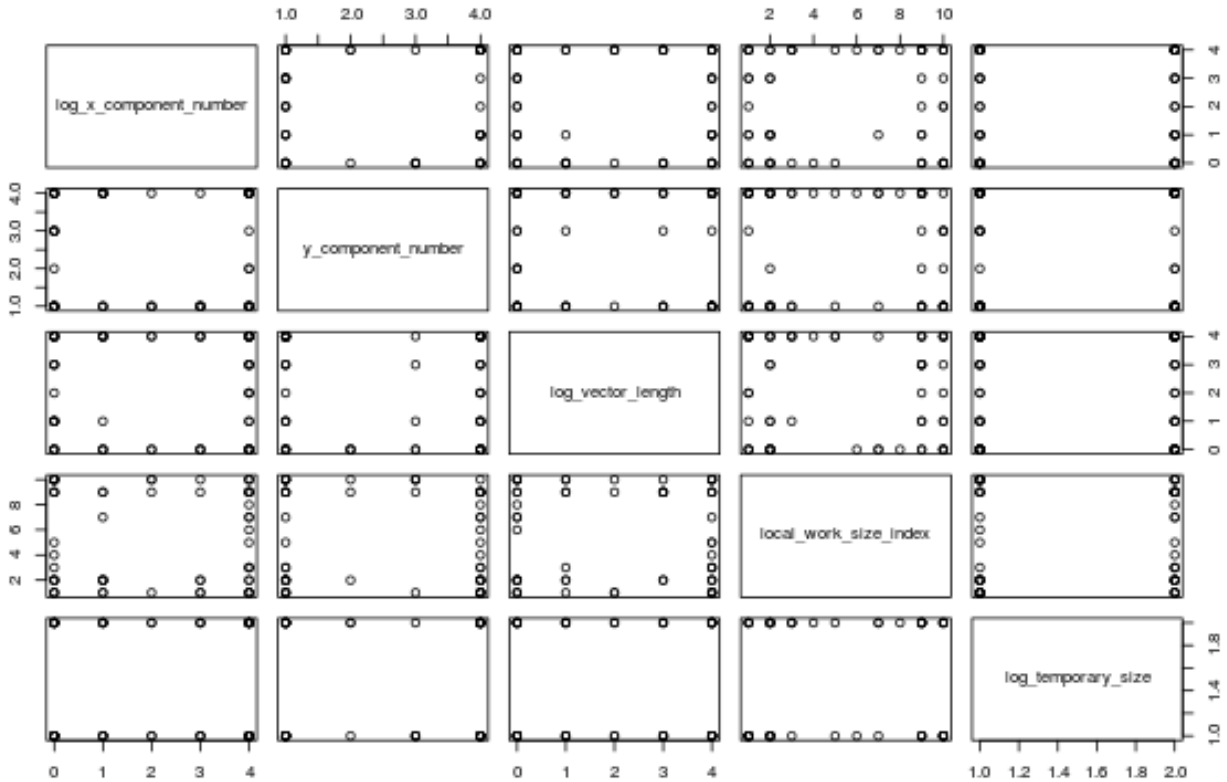
Local work size impact



Topic

Design of experiment to sample the search space

- Autotuning = experimenting
- Design of experiment to sample the search
- Extract information using less points as possible
- Use a D-Optimal design



D-Optimal Design - Revelant parameters

- 60 / 2000 points

```
Call:
lm.default(formula = time_per_pixel ~ x_component_number + y_component_number +
    vector_length, data = set)

Residuals:
       Min         1Q     Median         3Q        Max
-4.493e-10 -2.389e-10 -5.766e-11  2.031e-10  8.749e-10

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)         6.540e-10  1.194e-10   5.478 1.06e-06 ***
x_component_number -2.019e-11  6.648e-12  -3.037  0.00362 **
y_component_number -8.546e-11  3.202e-11  -2.669  0.00994 **
vector_length       4.126e-11  6.690e-12   6.167 8.11e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.449e-10 on 56 degrees of freedom
Multiple R-squared:  0.5265,Adjusted R-squared:  0.5011
F-statistic: 20.76 on 3 and 56 DF,  p-value: 3.617e-09
```

D-Optimal Design - Interactions

```
Call:
lm.default(formula = time_per_pixel ~ x_component_number * y_component_number *
    vector_length, data = set)

Residuals:
       Min         1Q     Median         3Q        Max
-5.829e-10 -7.267e-11 -1.375e-11  7.602e-11  5.229e-10

Coefficients:
                                        Estimate Std. Error t value Pr(>|t|)
(Intercept)                            2.574e-10  1.679e-10   1.533    0.131
x_component                            2.740e-11  1.705e-11   1.607    0.114
y_component                           -4.707e-11  5.665e-11  -0.831    0.410
vector_length                          9.656e-11  1.432e-11   6.743 1.27e-08 ***
x_component:y_component               -2.337e-12  5.615e-12  -0.416    0.679
x_component:vector_length             -7.727e-12  1.661e-12  -4.653 2.29e-05 ***
y_component:vector_length             -6.872e-12  5.127e-12  -1.340    0.186
x_component:y_component:vector_length  8.645e-13  5.453e-13   1.585    0.119
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.448e-10 on 52 degrees of freedom
Multiple R-squared:  0.7784,Adjusted R-squared:  0.7486
F-statistic: 26.1 on 7 and 52 DF,  p-value: 6.446e-15
```

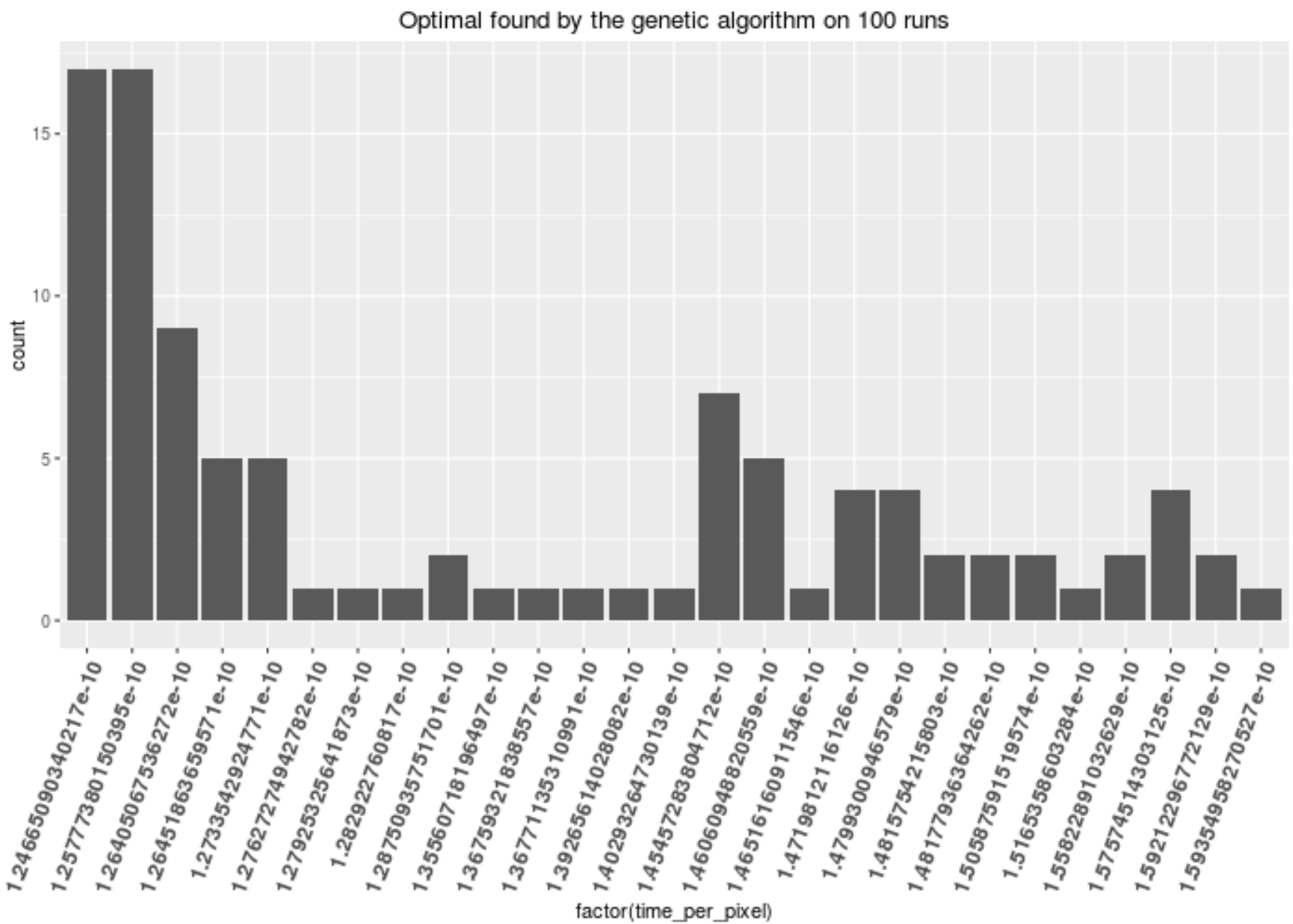
D-Optimal Design - A better model

```
Call:
lm.default(formula = time_per_pixel ~ x_component_number + y_component_number +
    vector_length + x_component_number:vector_length, data = set)

Residuals:
       Min          1Q      Median          3Q         Max
-5.388e-10 -8.124e-11  9.820e-12  7.812e-11  6.241e-10

Coefficients:
                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                       3.131e-10  9.776e-11   3.203  0.00226 **
x_component_number                2.108e-11  7.406e-12   2.847  0.00620 **
y_component_number               -6.970e-11  2.313e-11  -3.013  0.00390 **
vector_length                     8.010e-11  7.175e-12  11.163 9.36e-16 ***
x_component_number:vector_length -5.426e-12  7.435e-13  -7.297 1.23e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.481e-10 on 55 degrees of freedom
Multiple R-squared:  0.7594,Adjusted R-squared:  0.7419
F-statistic:  43.4 on 4 and 55 DF,  p-value: < 2.2e-16
```

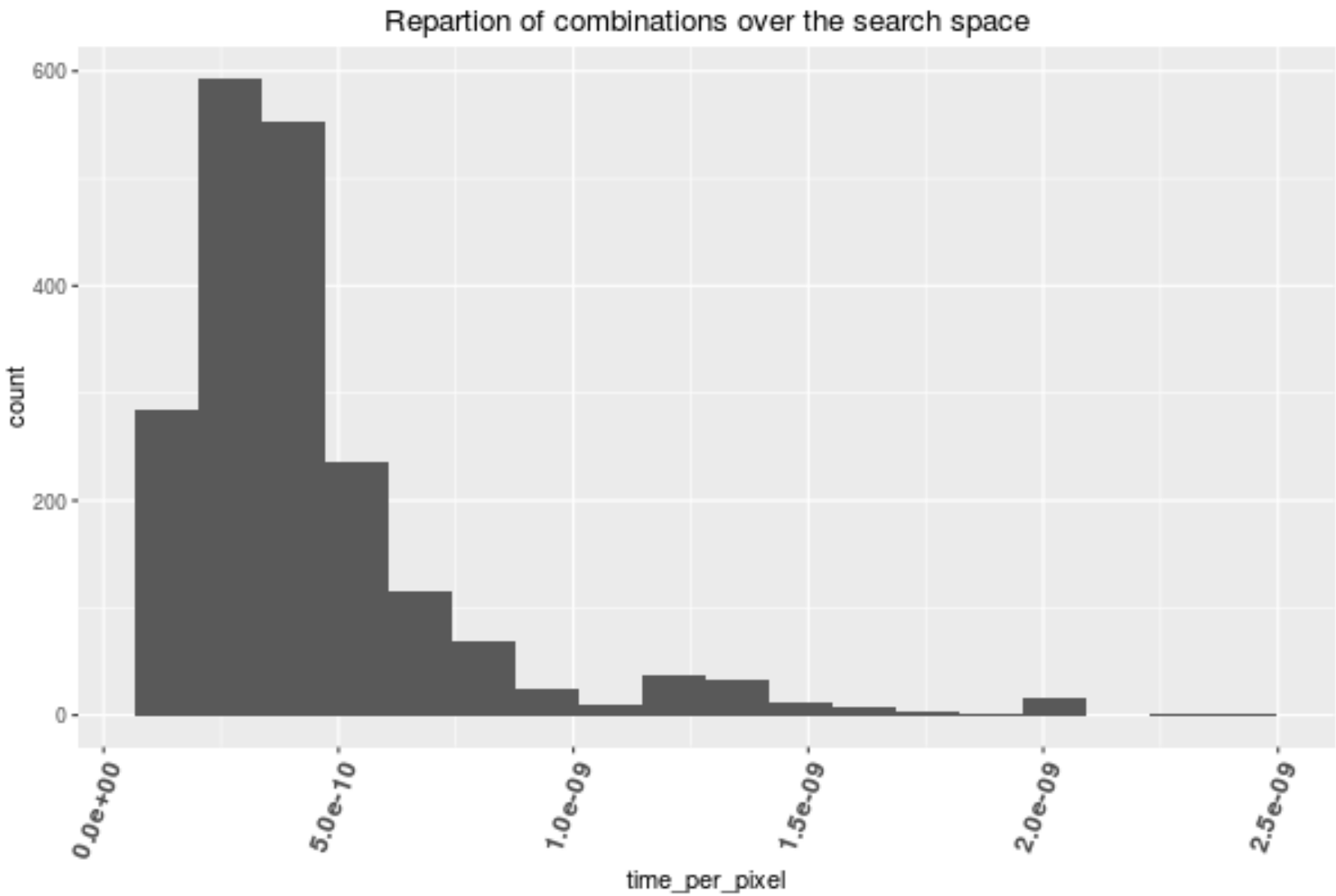
Quick look at the efficiency of the Genetic Algorithm?

- BOAST Implementation
- Population size 20
- Generation limit 100
- Warning probably not correctly tuned for this instance of problem

Quick look at the efficiency of the Genetic Algorithm?



Repartition of good combinations



Topic

How to map correctly the work on the GPU?

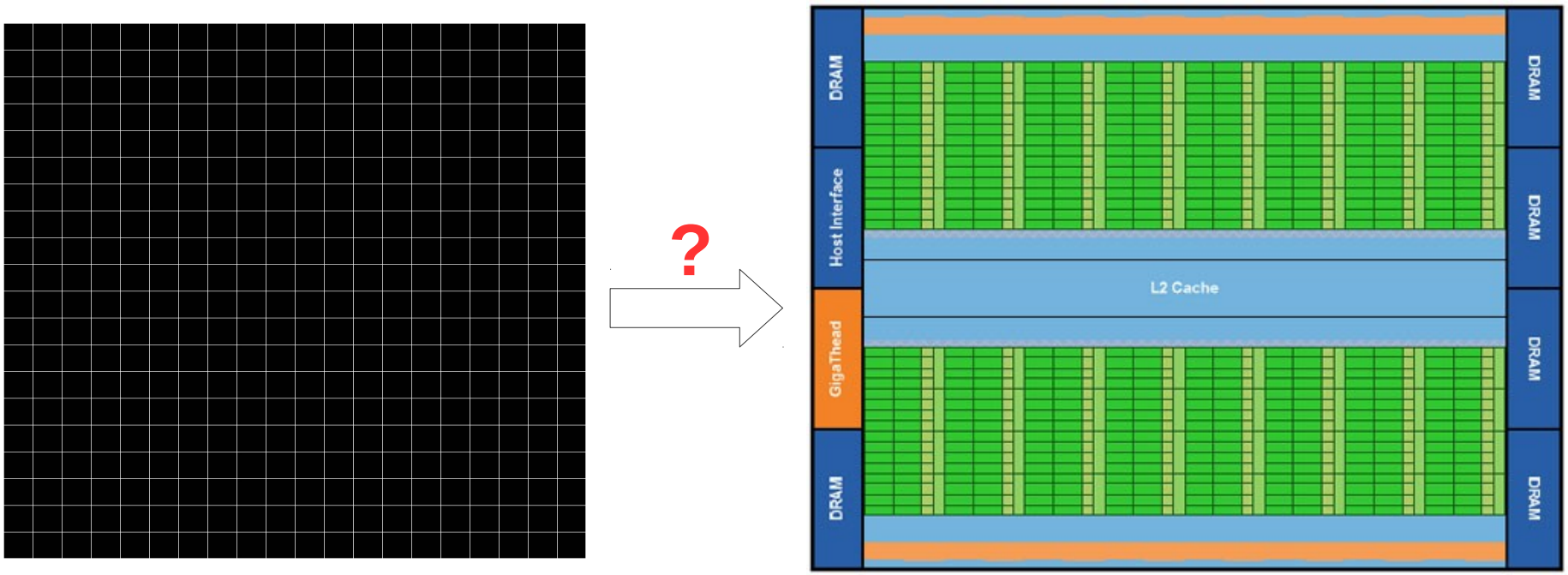


Figure: Gpu Mapping

How to map correctly the work on the GPU?

Efficient parallelism

- Keep computational units busy as much as possible → using enough threads to do the work → quantity of work per thread
- Threads are grouped by block → mapped on compute units → Shape and size of the group have an impact

Efficient memory use

- Grouping memory access and using cache efficiently to reduce idle time → quantity of work per thread, shape and size of the group

Trade-off

- Bigger blocks → more efficient data sharing
- Smaller blocks → better compute unit occupancy

Expressing the search space

Toward a more precise modelization

- The number of threads per blocks
- The shape of the blocks (x, y repartition)
- The amount of work per threads
- The shape of the work (x, y repartition)

Dealing with unfeasible points

- Limited by the size of the kernel $\rightarrow$ OUT_OF_RESSOURCES
  - Number of 32-bits registers $\rightarrow$ 65536
  - Shared memory $\rightarrow$ 48K
- Inaccurate combinations : E.g thread_number = 32 and lws_y = 64
- More advanced filtering $\rightarrow$ Need more expressivity

Constraints mechanism

- Ensure that all points respect the constraints
- Rules = set of boolean expressions
- All rules must be respected

```
:rules => [":lws_y <= :threads_number",
           ":threads_number % :lws_y == 0",
           ":elements_number >= :y_component_number",
           ":elements_number % :y_component_number == 0",
           ":elements_number / :y_component_number <= 4",
           ":elements_number * :vector_length * :temporary_size *
                               :threads_number <= 1024 * 53"
         ]
```

Topic

Search space formulation

- Comparing 2 formulations:
  - Threads organization:
    - number of threads and number of threads on y-axis $\rightarrow$ hierarchy
    - number of threads on x and y-axis
- Which is better to modelize $\rightarrow$ on which D-Optimal is more efficient?

More complex search space and D-Optimal

- Bigger work space $\rightarrow$ 19040 combinations
- Adding more informations:
  - Performance of all runs, not only the min $\rightarrow$ warmup effect
    - How to make good estimate of the min
- Validation against bruteforce and comparing with genetic algorithm (efficient on large space)
- How to determine how many points we need with D-Optimal?
- How to handle categorical variables in the linear model?
- How to handle constraints, discrete and categorical variables in the design of experiment?

Summary

- Using a simple linear regression allowed us to extract informations about the search space
- Just few points are need with a D-optimal design (3% of the search space)
- The better the described the search space, the better the model $\rightarrow$ check the importance
- Handle constraints, discrete and categorical variables in the analysis
- Need to validate with bigger, more complexe search space, different problem and architecture

End

Thank you for your attention.