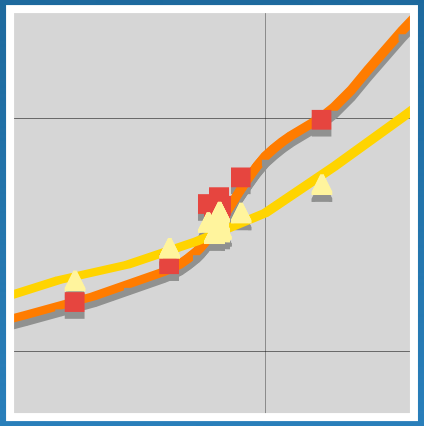
Design optimisation with Radiance

Giulio Antonutto Andrew Mc Neil Kristina Shea



ArupLighting

Summary

Optimisation?!

The tools

Combining them together

Proof of concept

Case studies

Optimisation: what's for?

Generate the optimal project configuration in order to maximise or minimise a given set of parameter such as:

- ☞ Daylight factor / irradiance / light levels
- ☞ Uniformity / distribution
- ☞ Light flow / symmetry of light
- ☞ Planning and massing of sites
- ☞ Percentage of shade

...

The tools

Mac OS X	Software recipient
Radiance	Calculation
Matlab / Octave	Optimisation
OpenDX	Visualisation

Mac OS X

- Commercial, true UNIX system with an excellent GUI
- Easy to set up and maintain
- Compatible with commercial applications
- Compatible with Open-source software
- Include excellent development tools

http://www.apple.com

Radiance

- Reference software for light simulations
- Highly scriptable and customisable
- Works on UNIX and therefore Mac OS X
- Free!

http://radsite.lbl.gov/radiance/

Radiance
Synthetic Imaging System

Matlab

- Matlab is a standard
- Easy to use, rich in documentation
- Generates fast code
- Can batch shell scripts and read/write files
- Works on Mac OS X

http://www.mathworks.com/

Octave

- Matlab compatible environment
- Can batch shell scripts and read/write files
- Works on UNIX and therefore Mac OS X
- Free!

http://www.octave.org/

OpenDX

- Powerful data visualisation software
- Works on UNIX and therefore Mac OS X
- Scriptable
- Free!

http://www.opendx.org/

OpenDX

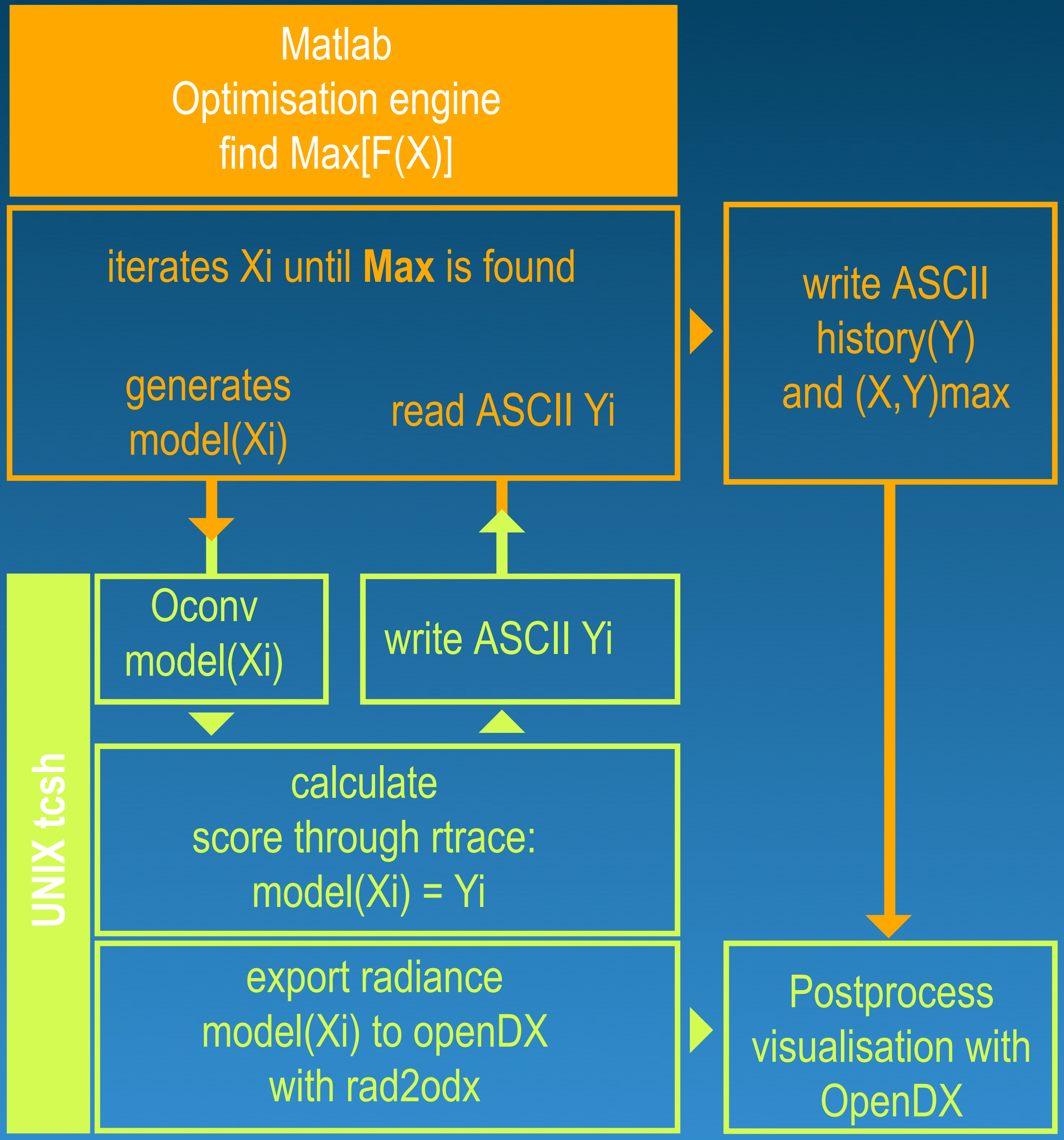
Combining them together

I. Matlab is used to create parametrically defined geometries

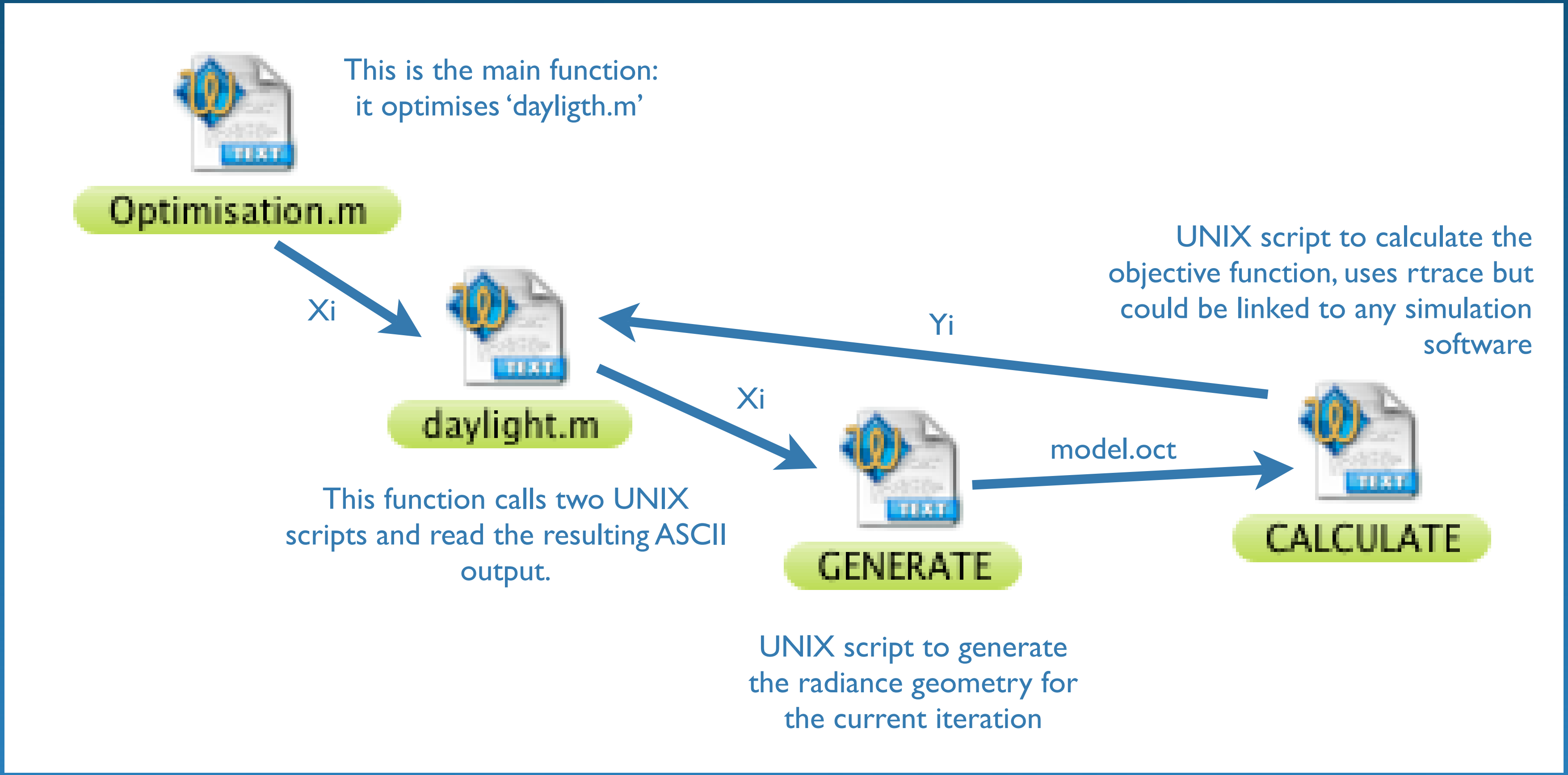
II. Radiance runs within Matlab as a shell command

III. Radiance results are exchanged with Matlab through ASCII text files

IV. Iteration are collected and post processed by OpenDX


Matlab
Optimisation engine
find Max[F(X)]
iterates Xi until **Max** is found
generates model(Xi)
read ASCII Yi
write ASCII history(Y) and (X,Y)max
Oconv model(Xi)
write ASCII Yi
calculate score through rtrace: model(Xi) = Yi
export radiance model(Xi) to openDX with rad2odx
Postprocess visualisation with OpenDX
UNIX tcsh

Workflow schematic



Proofs of concept

Example I - positioning luminaires

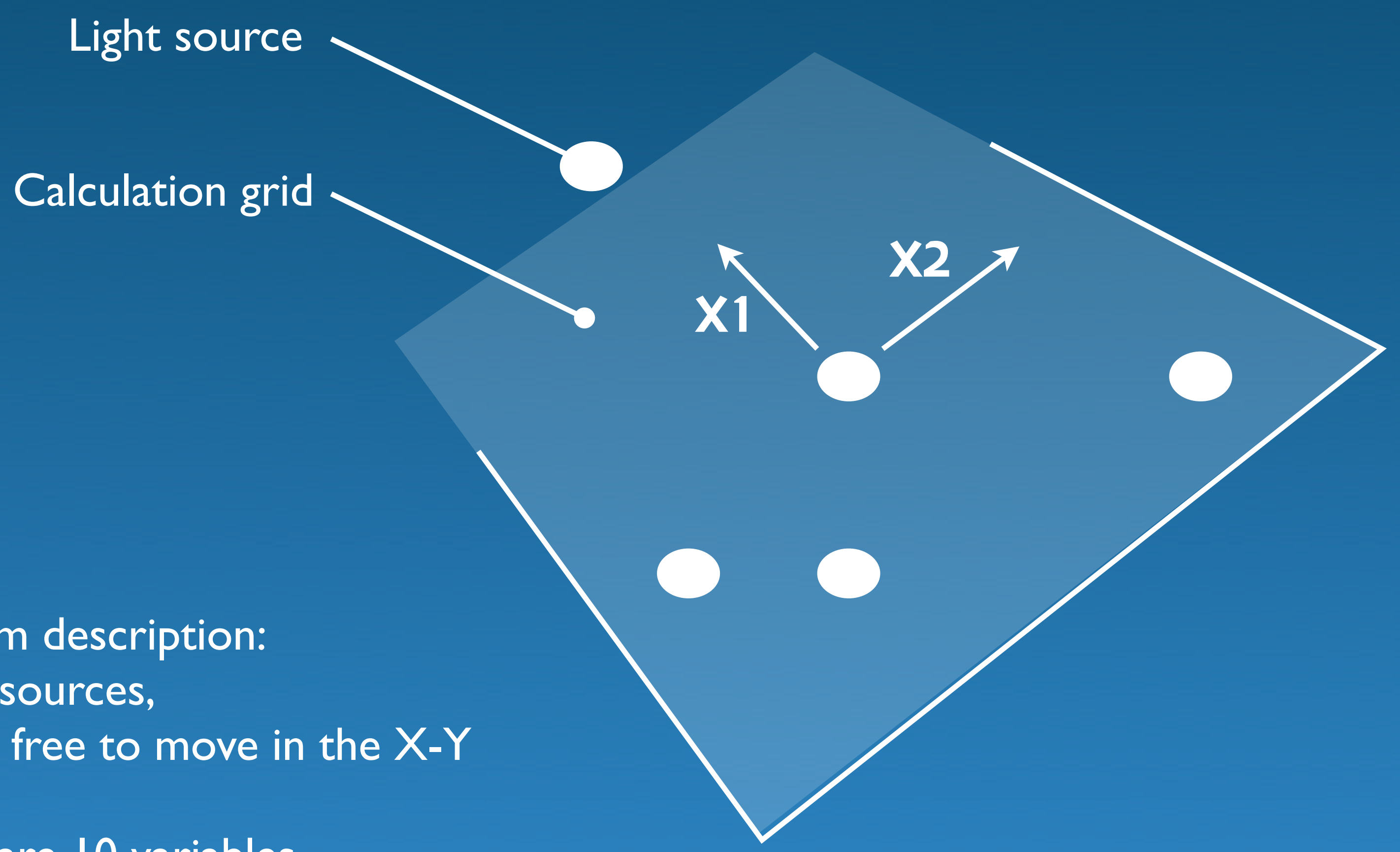
Example II - aiming luminaires

Example III - shaping a louvre system

Example IV - shaping a window

Example V - Sunlight and daylight availability

Example I

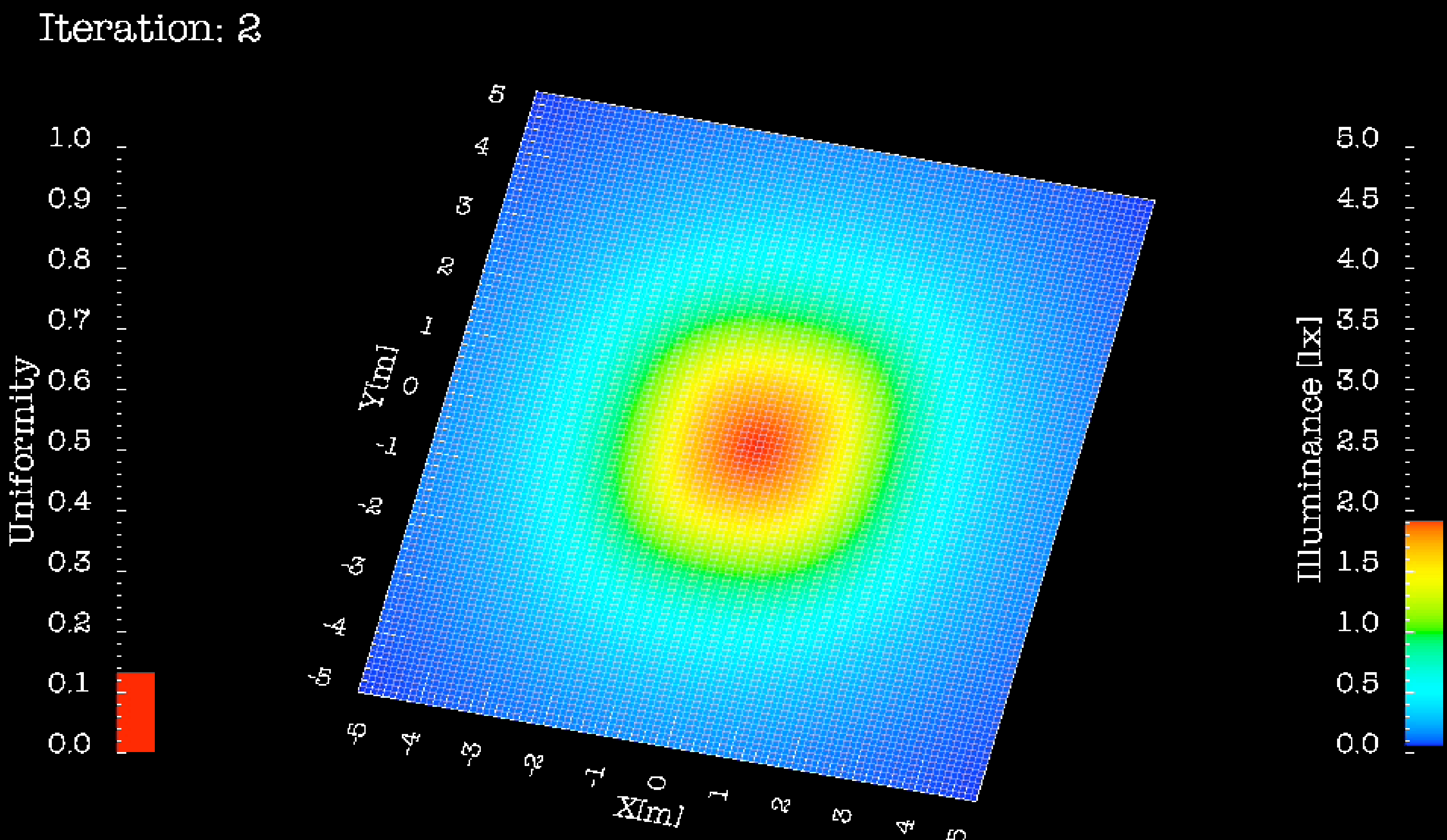


Problem description:
5 light sources,
Each is free to move in the X-Y plan.
There are 10 variables.
Illuminance and uniformity are maximised over a grid.
Starting position has all 5 sources in (0,0).

Objective function is: obj=(-1/uniformity)

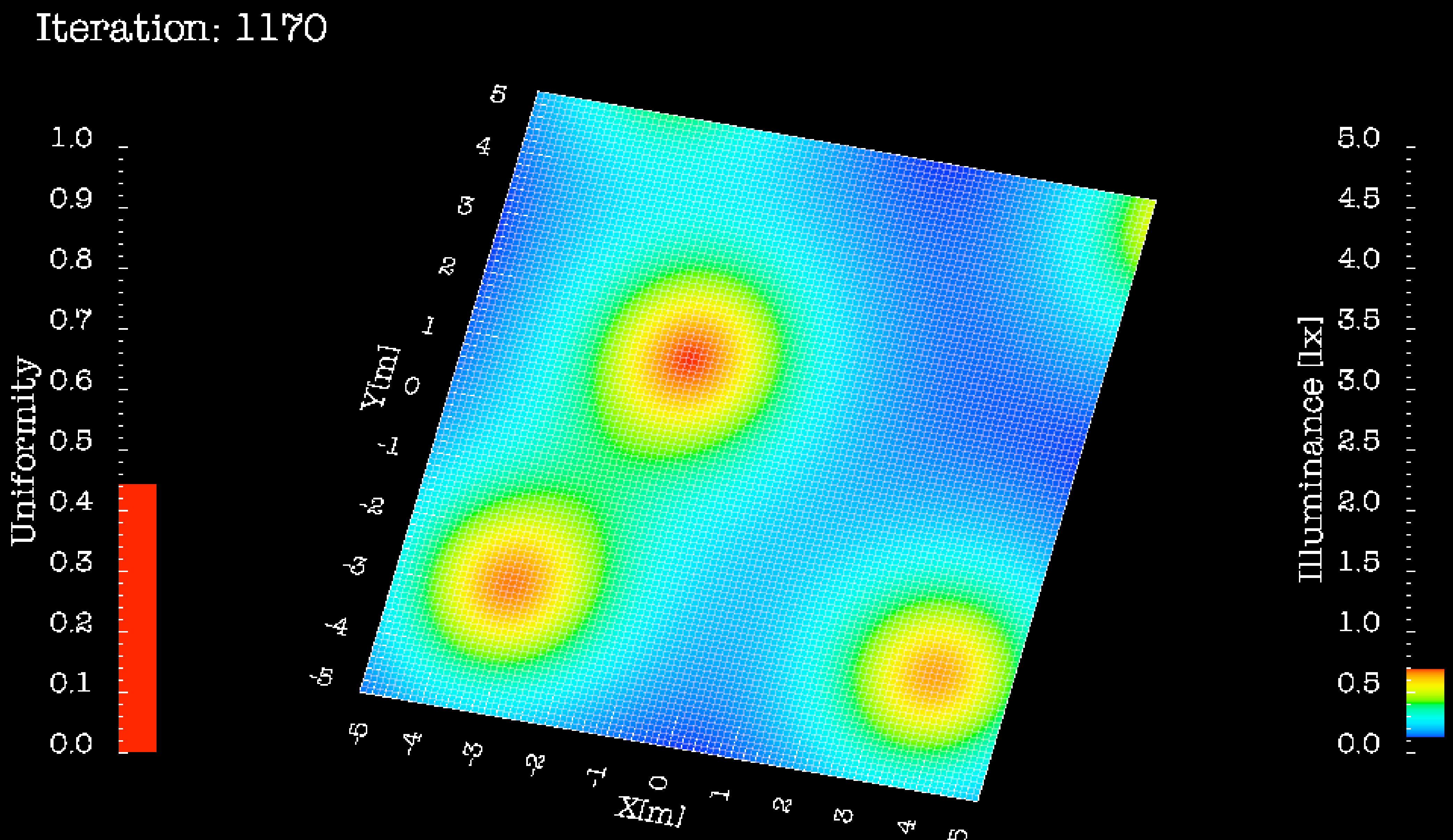
Optimisation engine: simplex search

Example 1



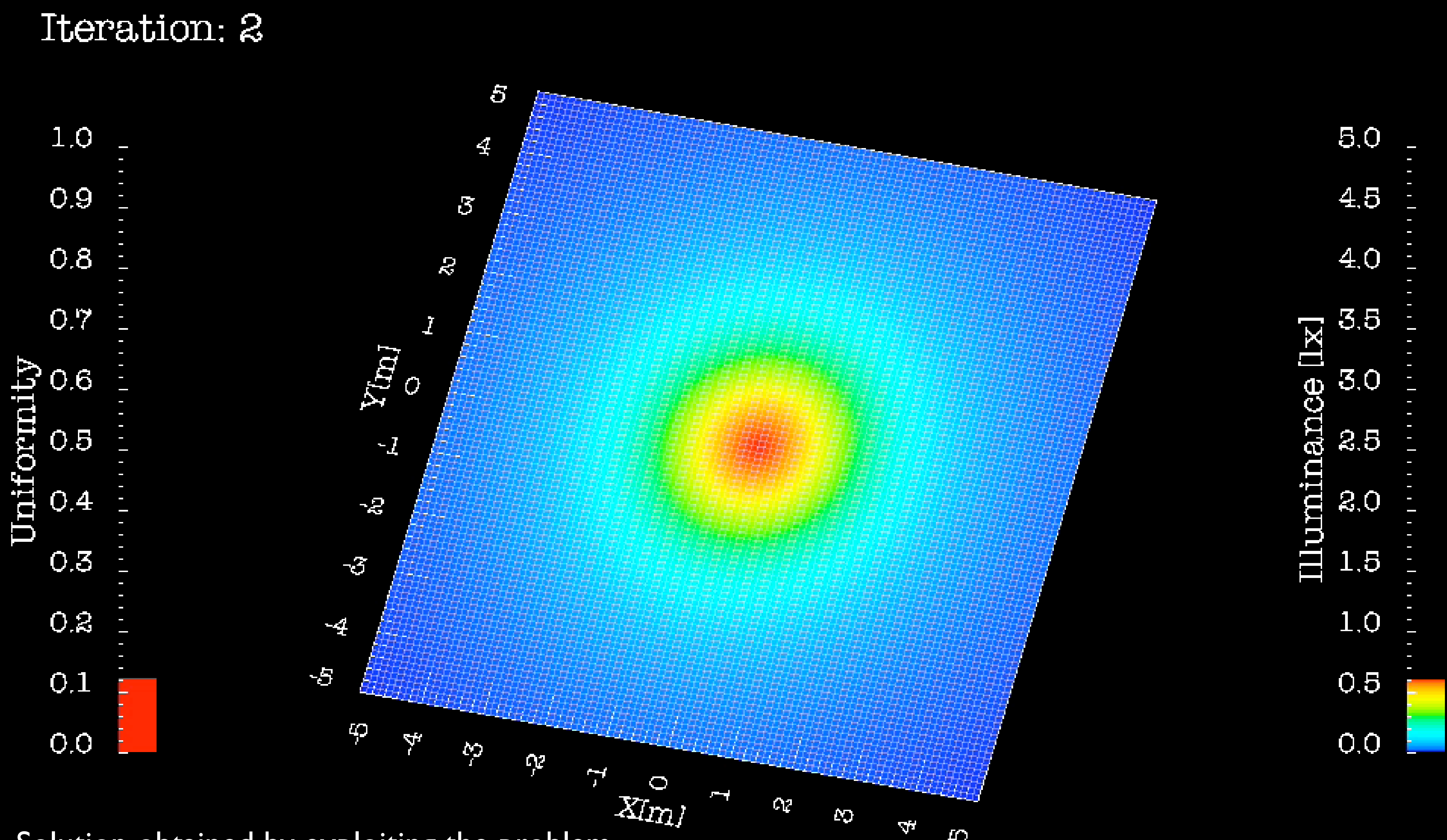
Iteration: 2
Uniformity
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Y[m]
5
4
3
2
1
0
-1
-2
-3
-4
-5
X[m]
-5
-4
-3
-2
-1
0
1
2
3
4
5
Illuminance [lx]
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0

Example I



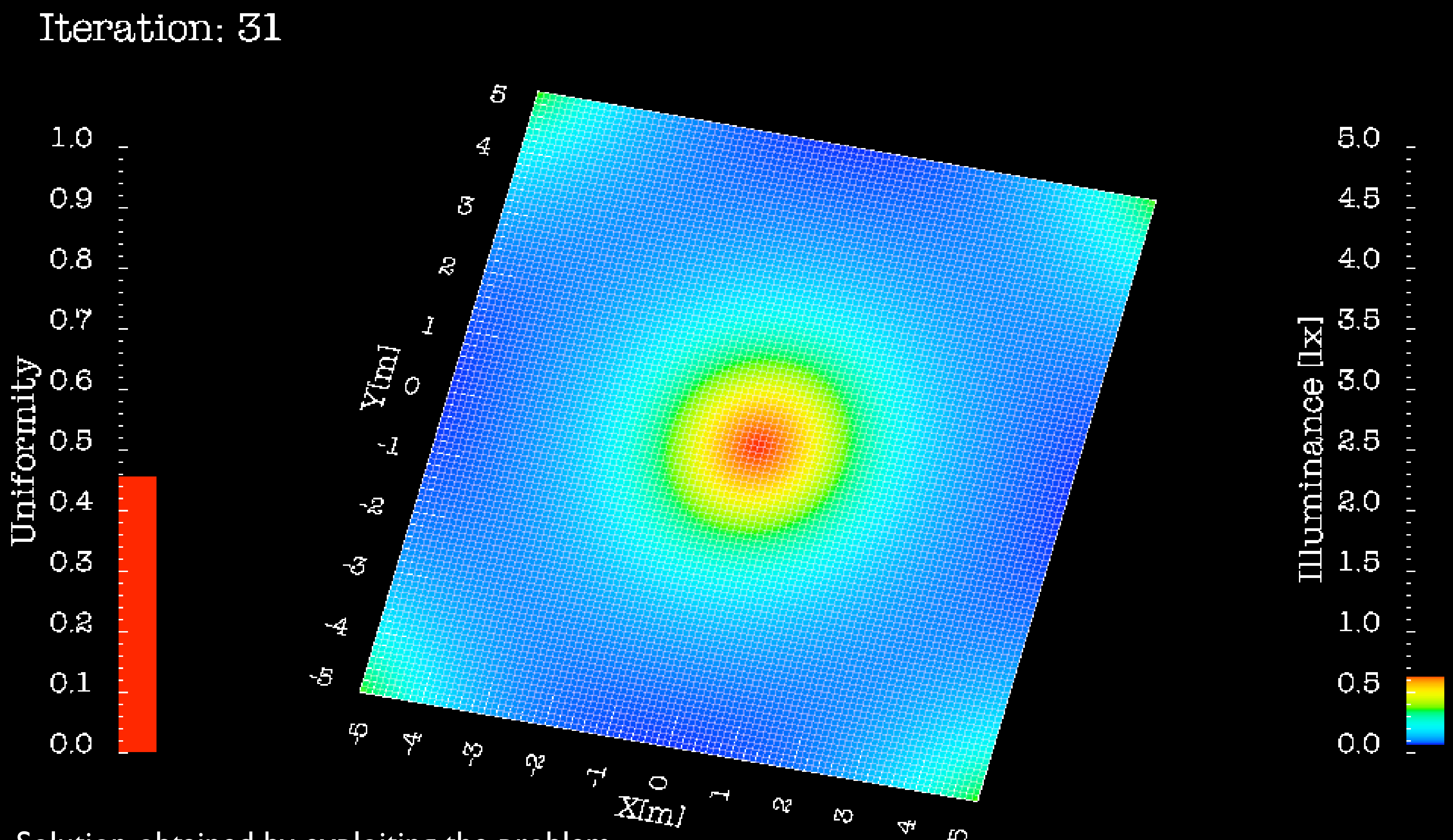
Full 10 variables solution.

Example 1



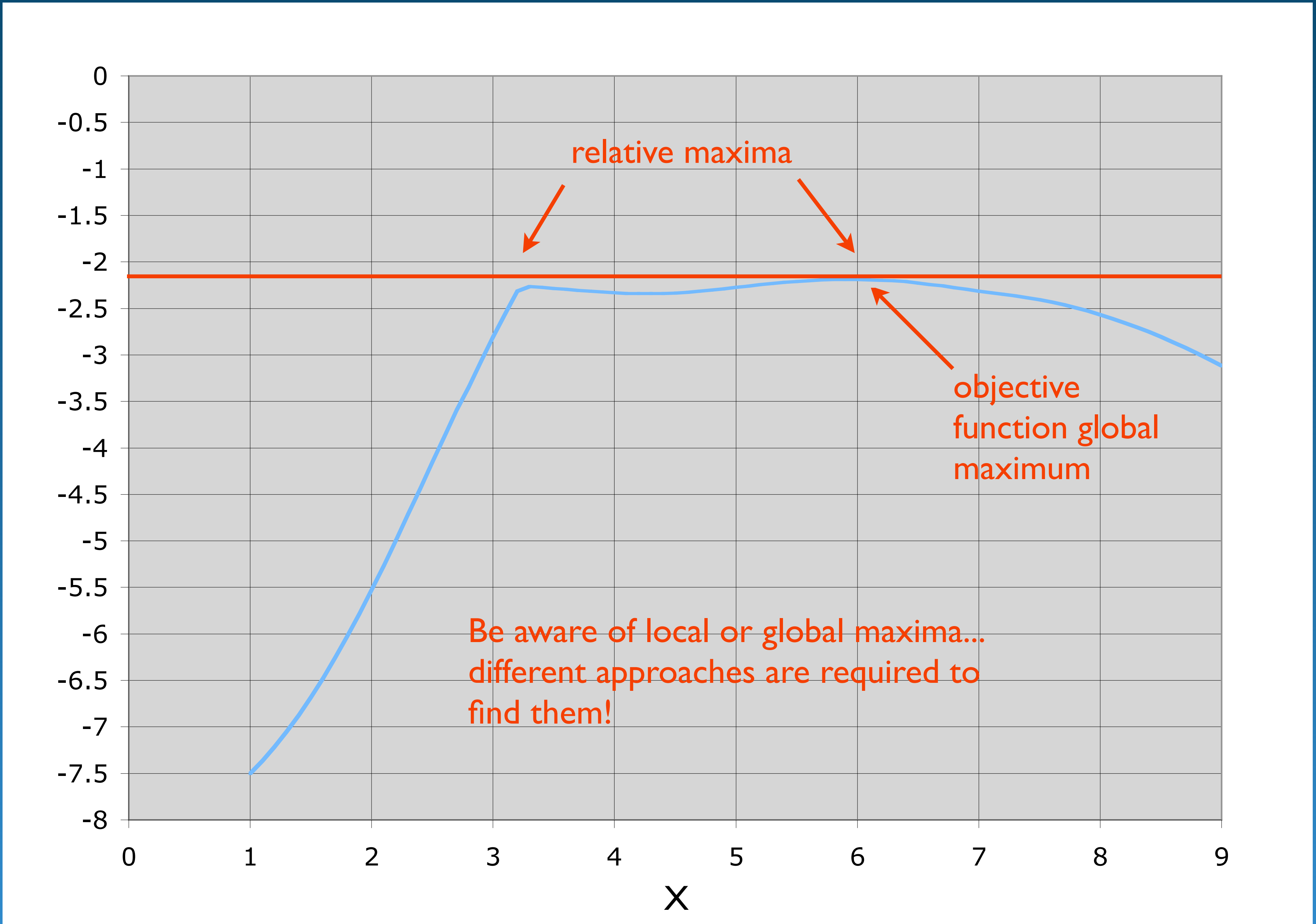
Solution obtained by exploiting the problem symmetry to reduce the variable number from 10 to 1.

Example 1

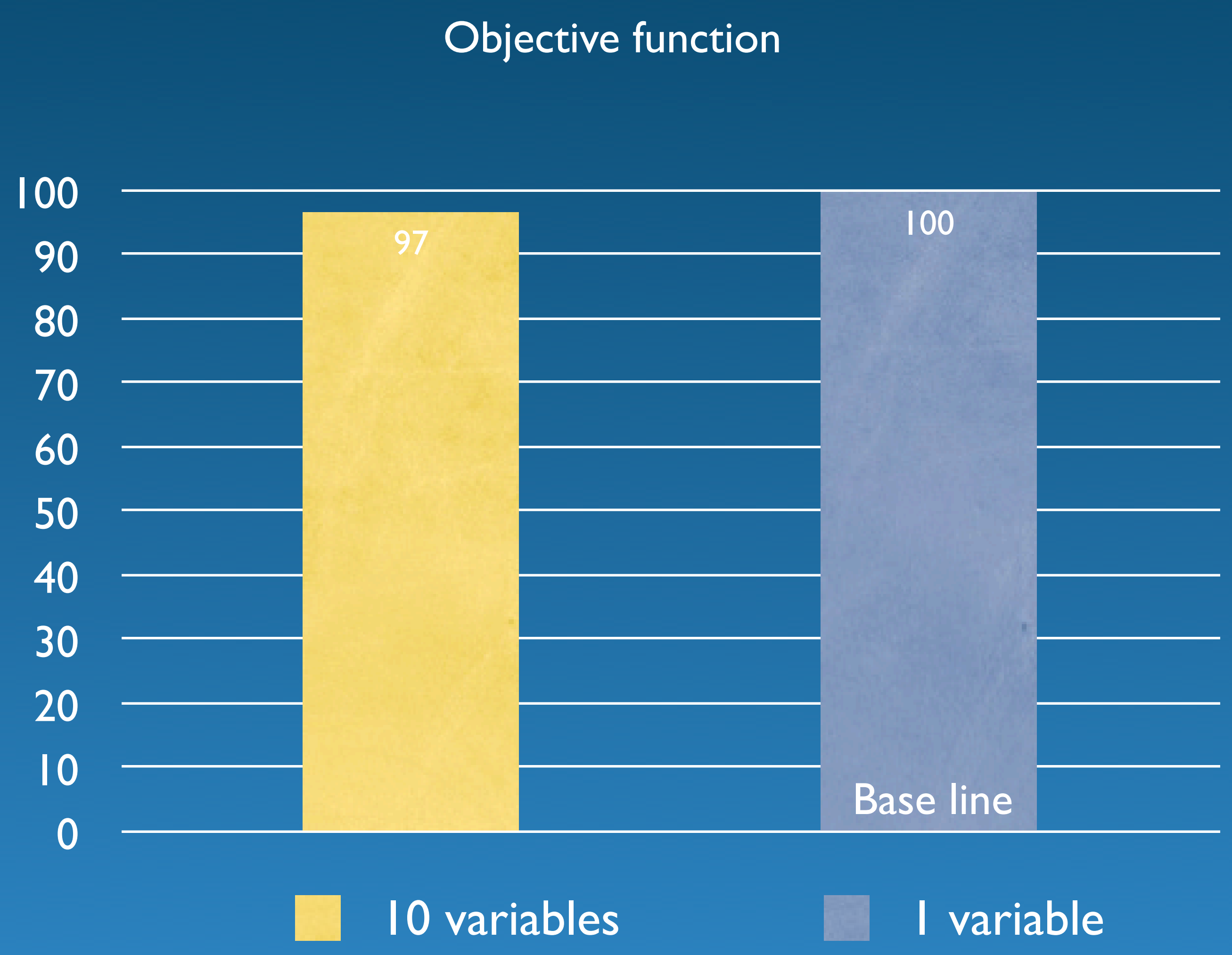


Solution obtained by exploiting the problem symmetry to reduce the variable number from 10 to 1.

Example I

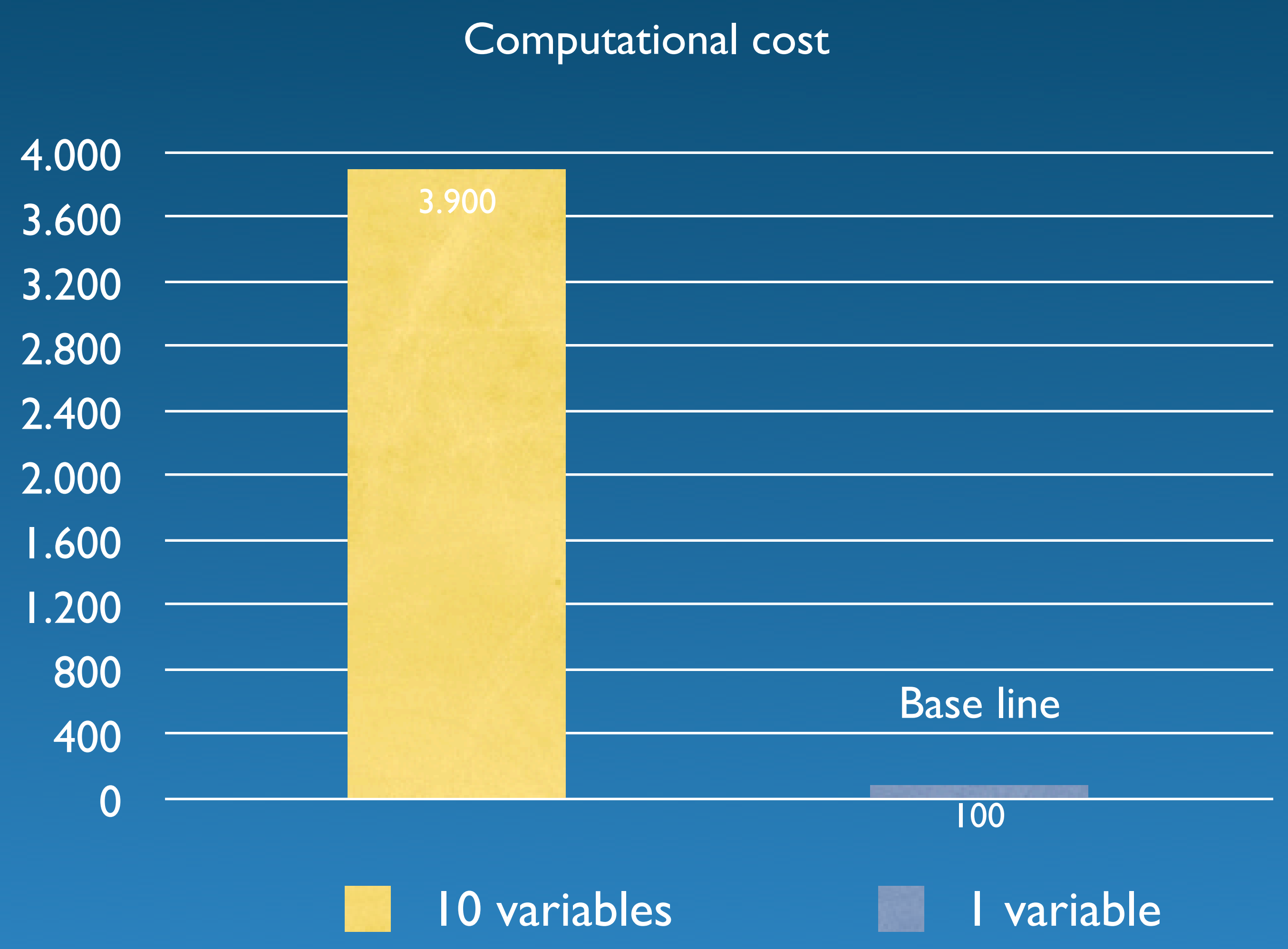

0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
-4
-4.5
-5
-5.5
-6
-6.5
-7
-7.5
-8
0
1
2
3
4
5
6
7
8
9
x
relative maxima
objective function global maximum
Be aware of local or global maxima... different approaches are required to find them!

Example I



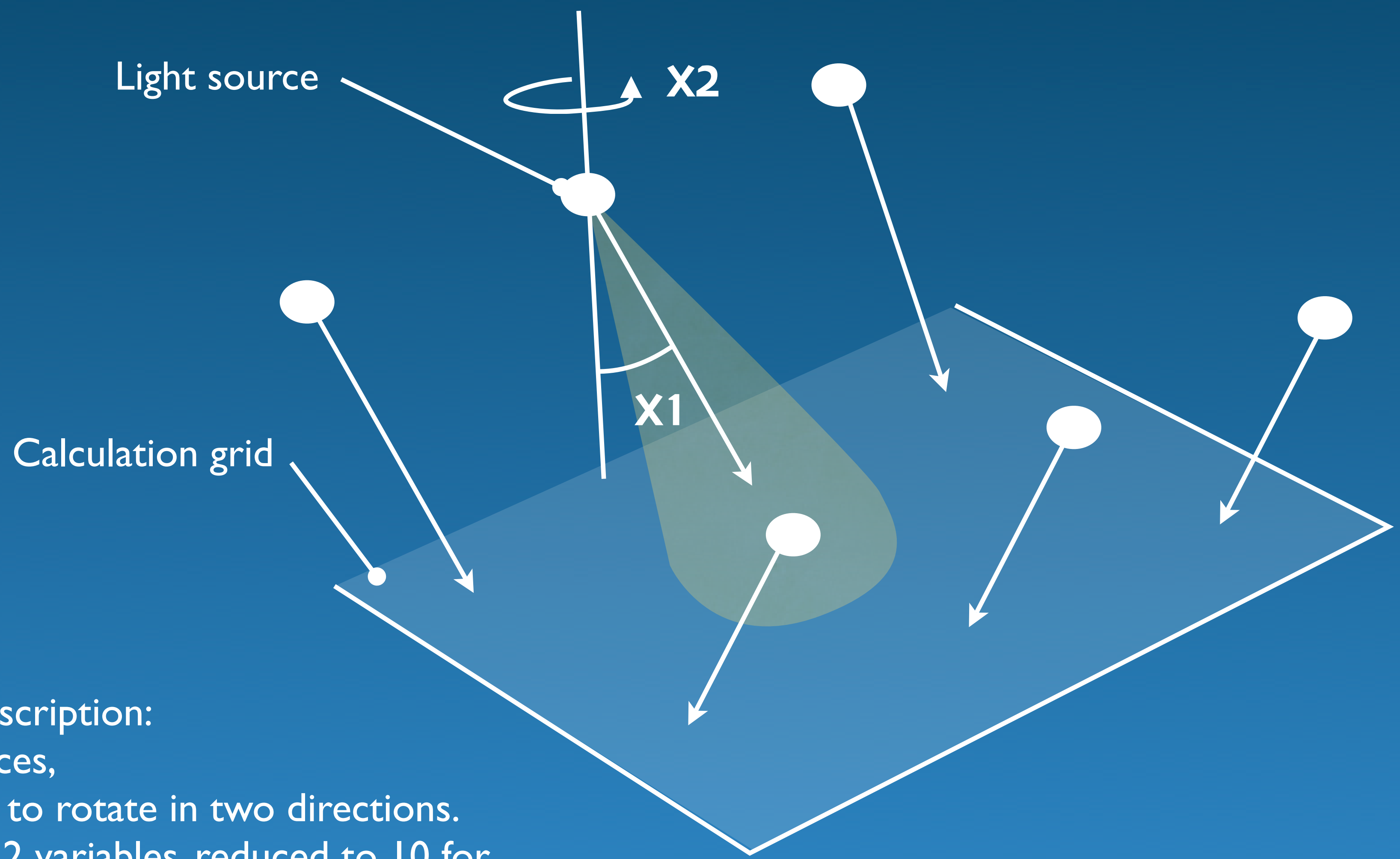
Comparison of optimal solution found: the difference is 3%.

Example I



Computational cost is 40x more for the 10 variables problem.

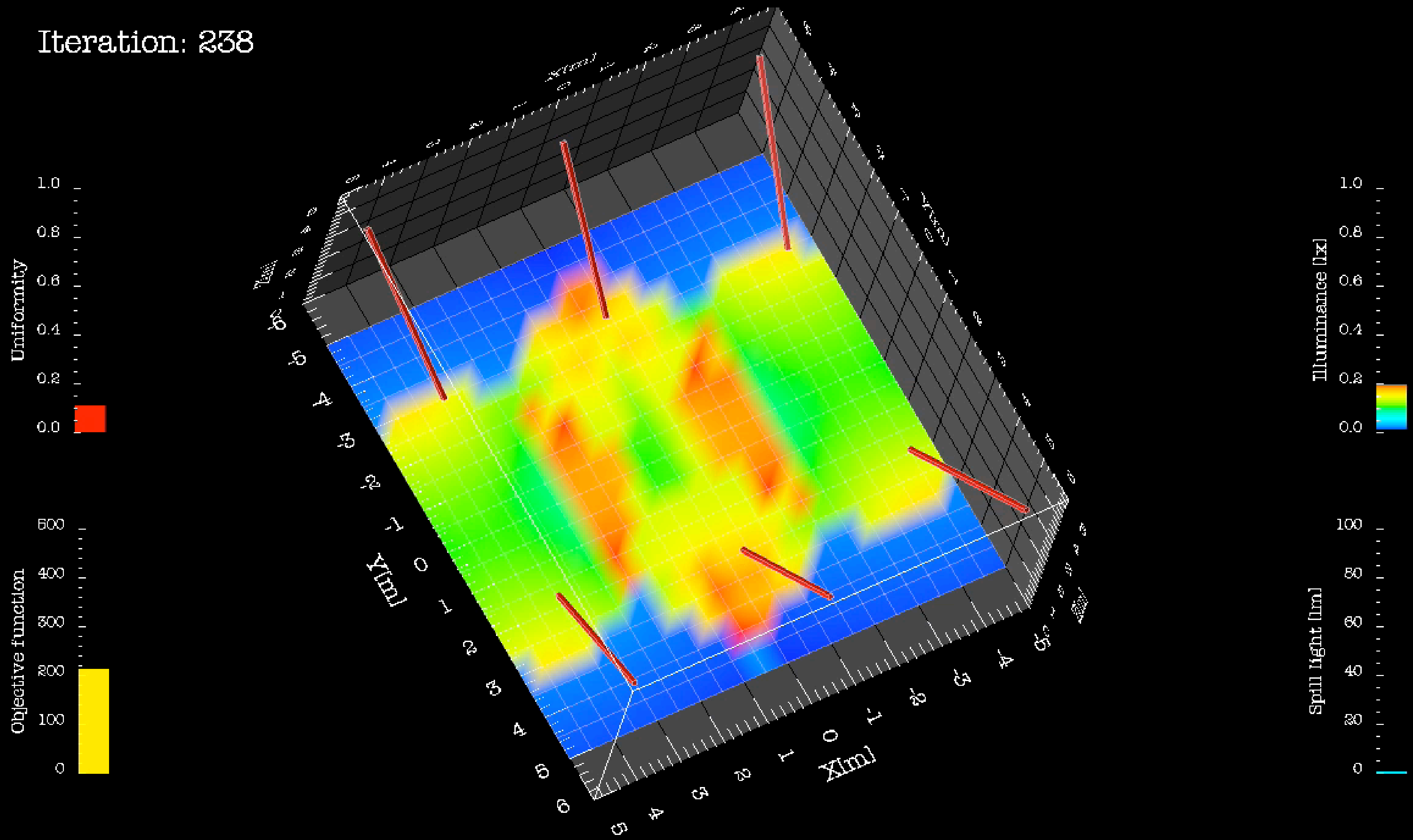
Example II



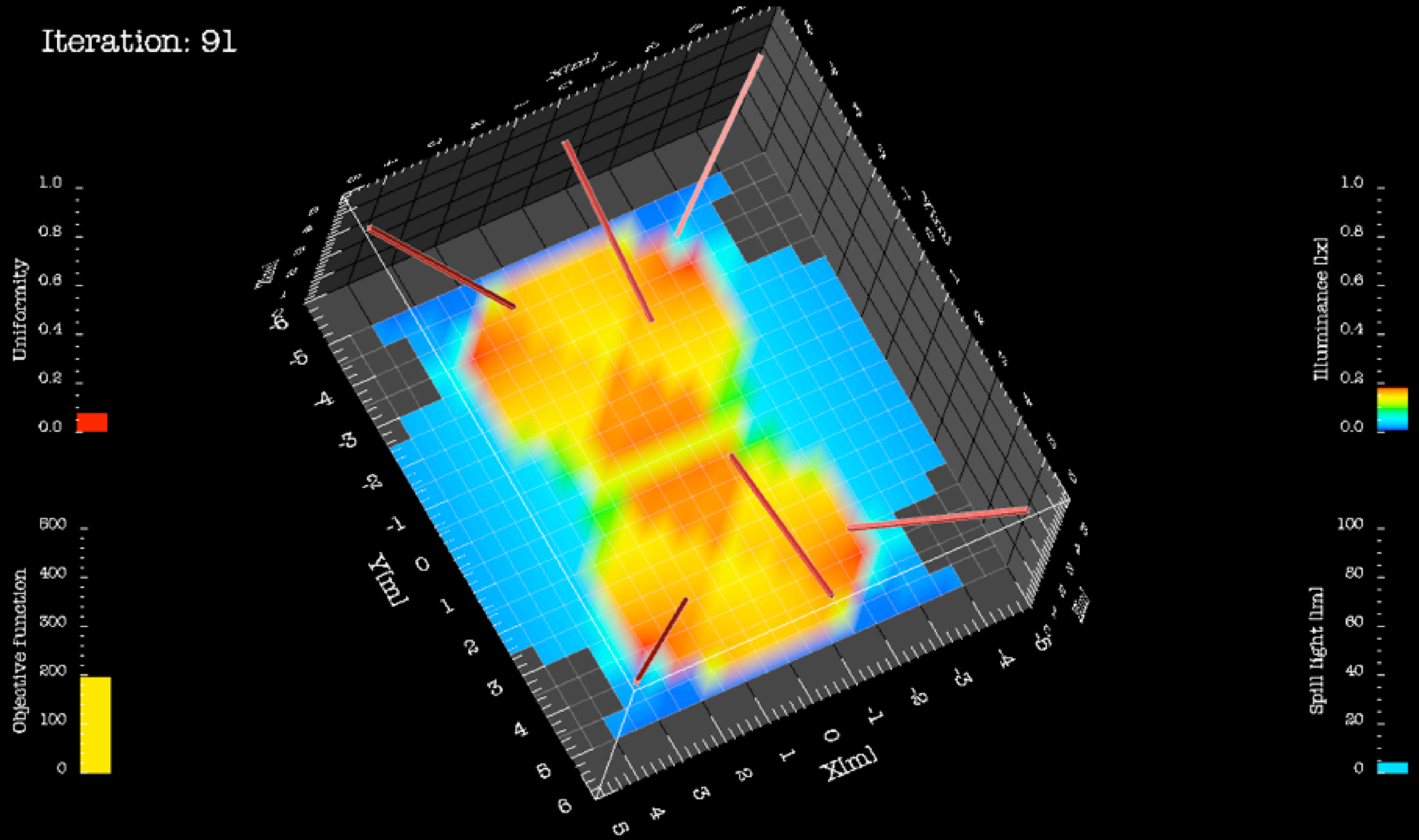
Problem description:
6 light sources,
Each is free to rotate in two directions.
There are 12 variables, reduced to 10 for symmetry (could have been reduced to 3).
Illuminance, uniformity are maximised over a grid.
Spill light is minimised.
Starting position has all 6 sources in pointing down.

Optimisation engine: simplex search

Example II

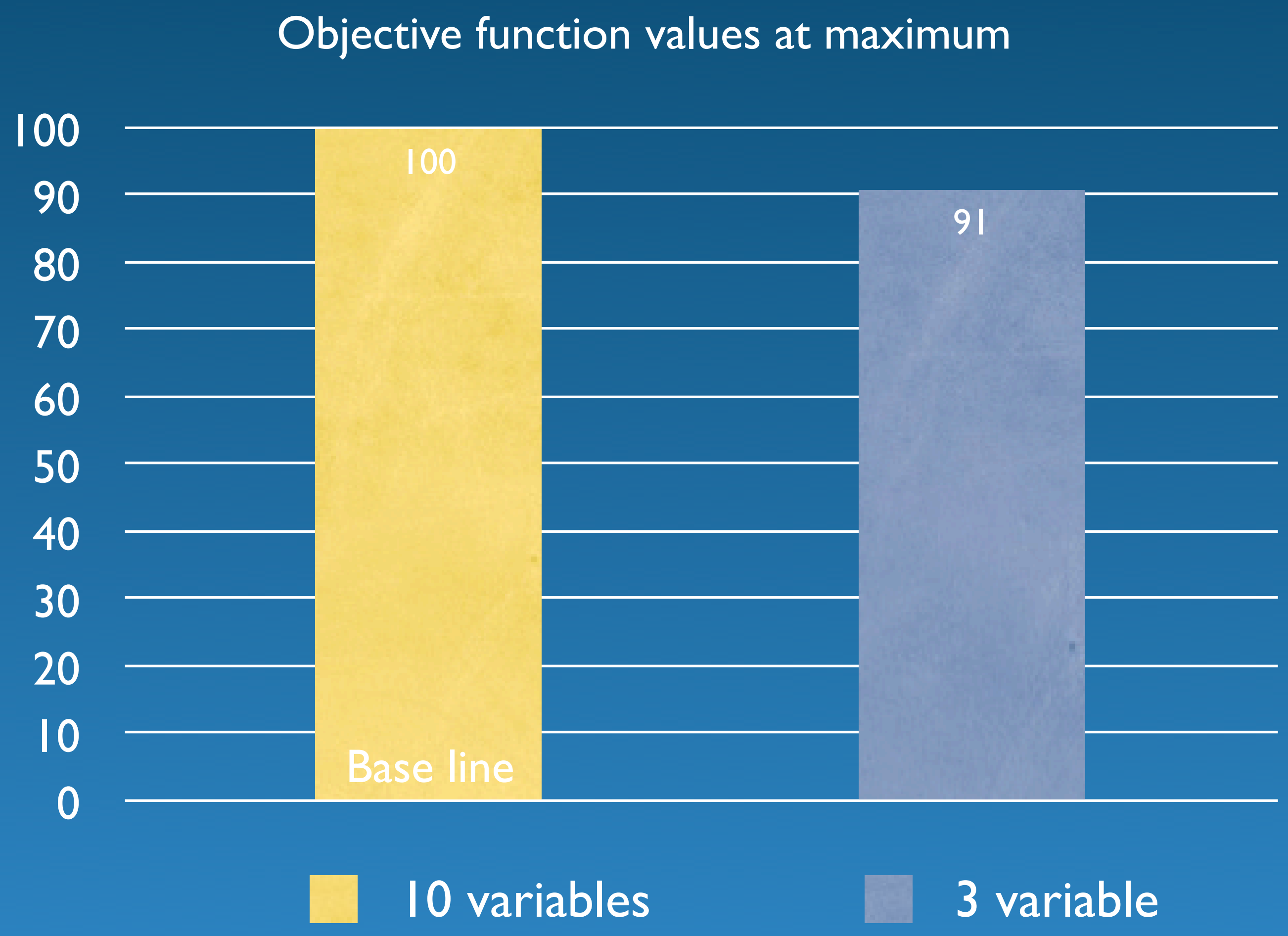

Iteration: 238
Uniformity
1.0
0.8
0.6
0.4
0.2
0.0
Objective function
500
400
300
200
100
0
Illuminance [lx]
1.0
0.8
0.6
0.4
0.2
0.0
Spill light [lm]
100
80
60
40
20
0
Y[m]
X[m]

Example II



Reduced problem, variables are now just 3 for symmetry.
Solution is marginally sub optimal but computational effort is 40% of the full problem.

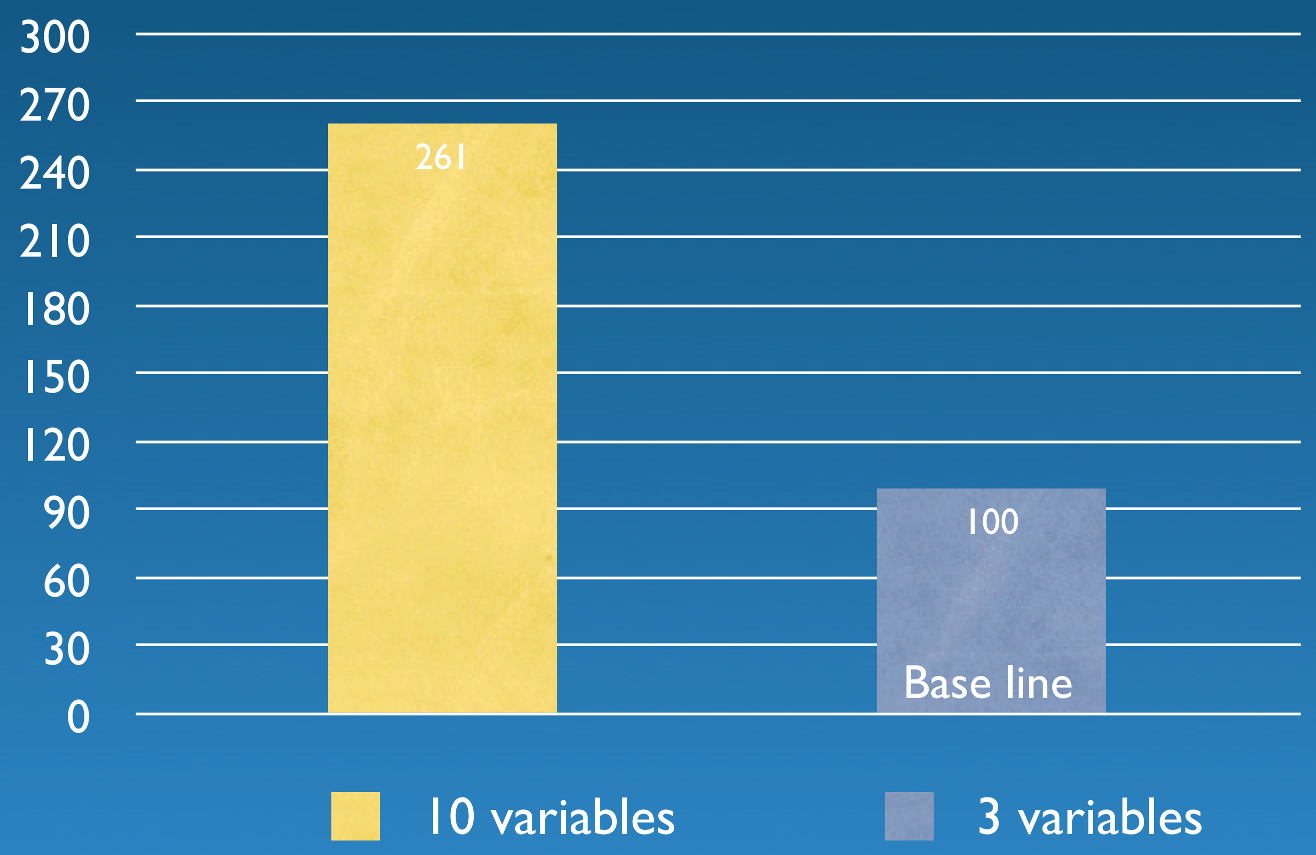
Example II



Comparison of optimal solution found: the difference is 9%.

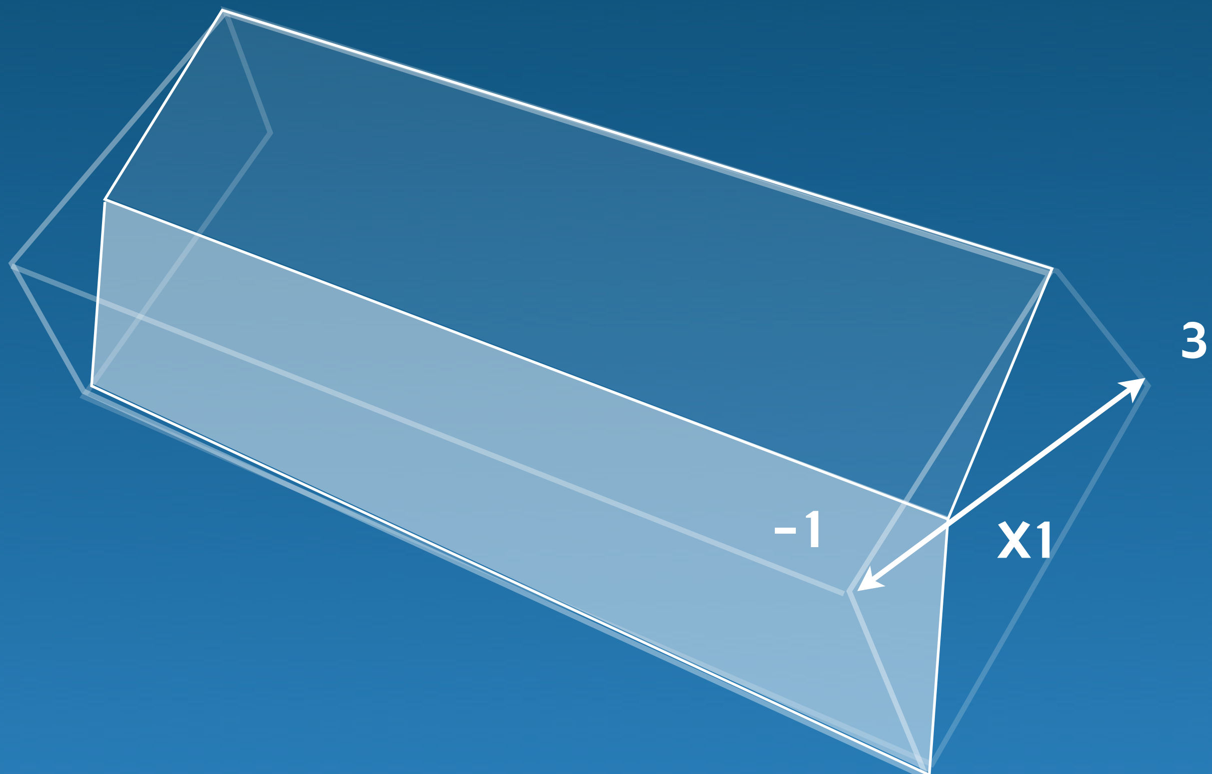
Example II

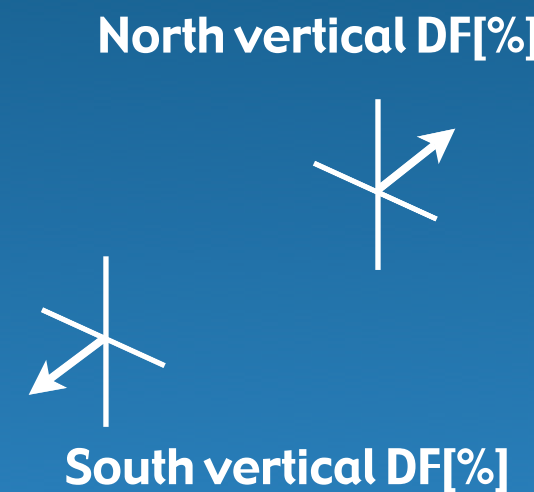
Computational cost:



Computational cost is 2.6x more for the 10 variables problem.

Example III



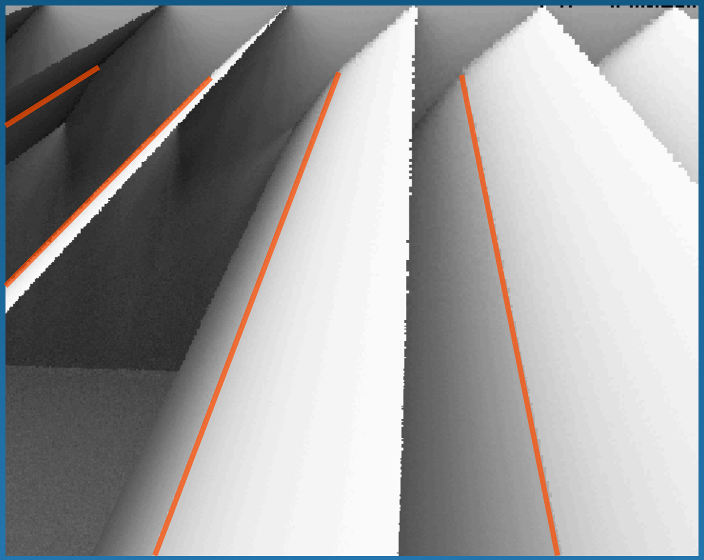


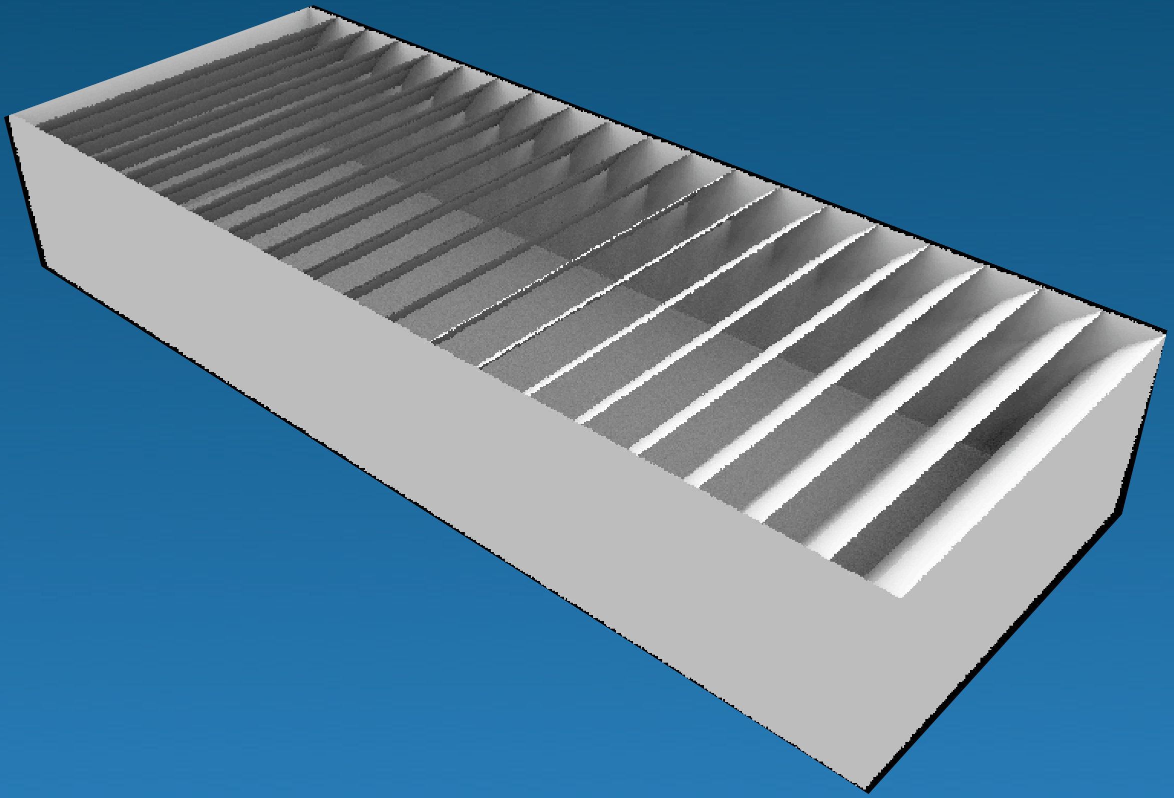
Problem description:

Optimisation of a North light louver system shape by adjusting the position of a control point, X, in a given interval. Optimisation program minimise the difference in N-S Daylight factor.

Optimisation engine:
simplex search with boundary restrictions

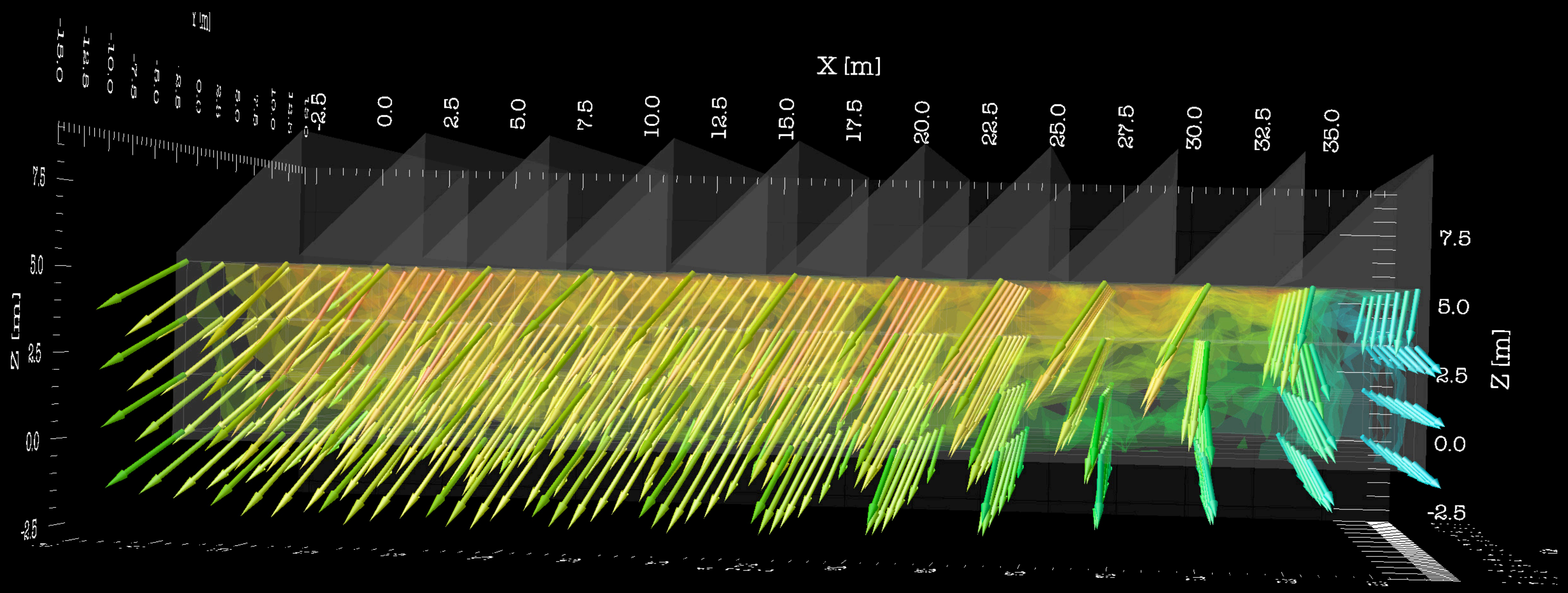
Example III





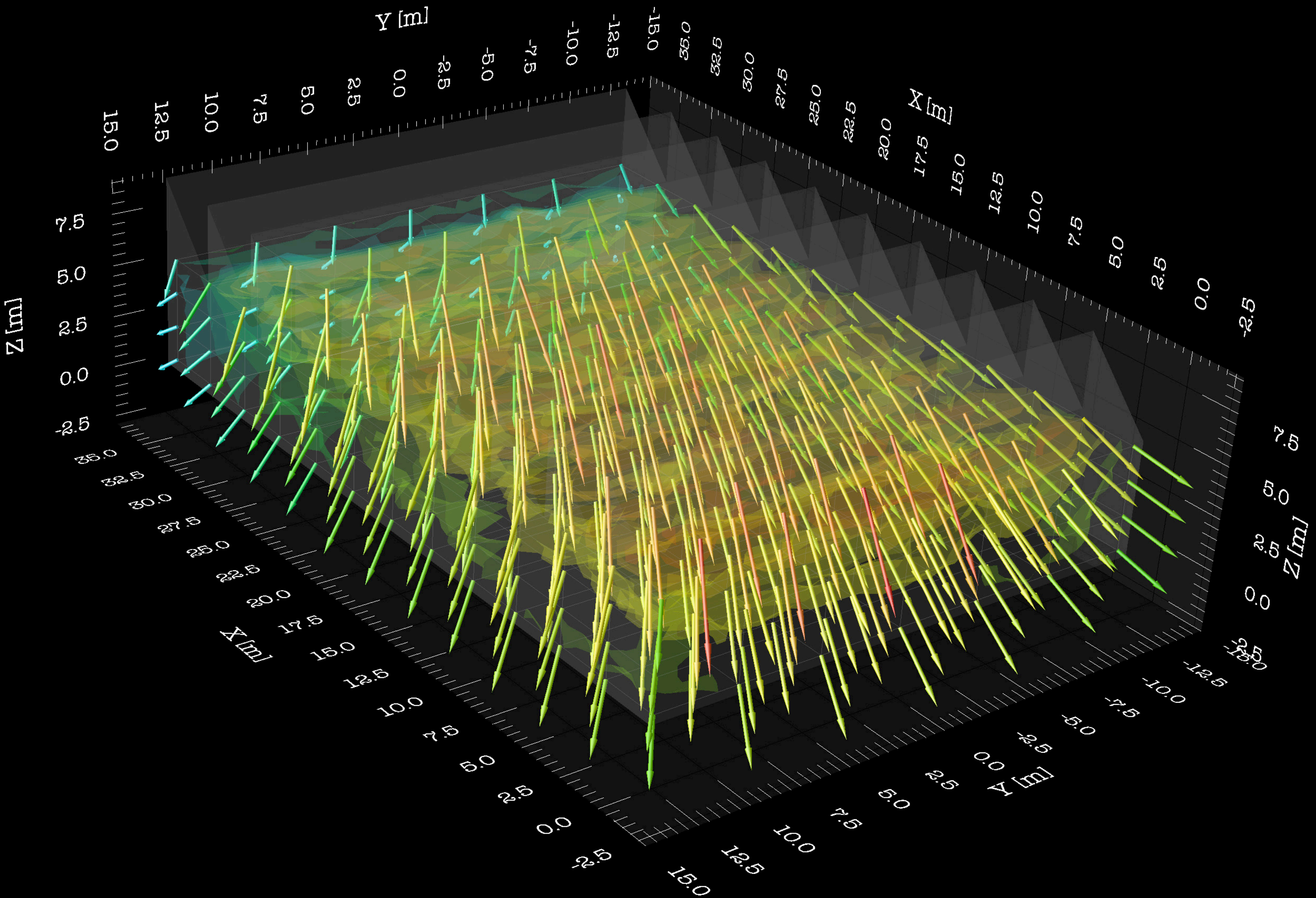
Radiance views of the louvered roof

Example III



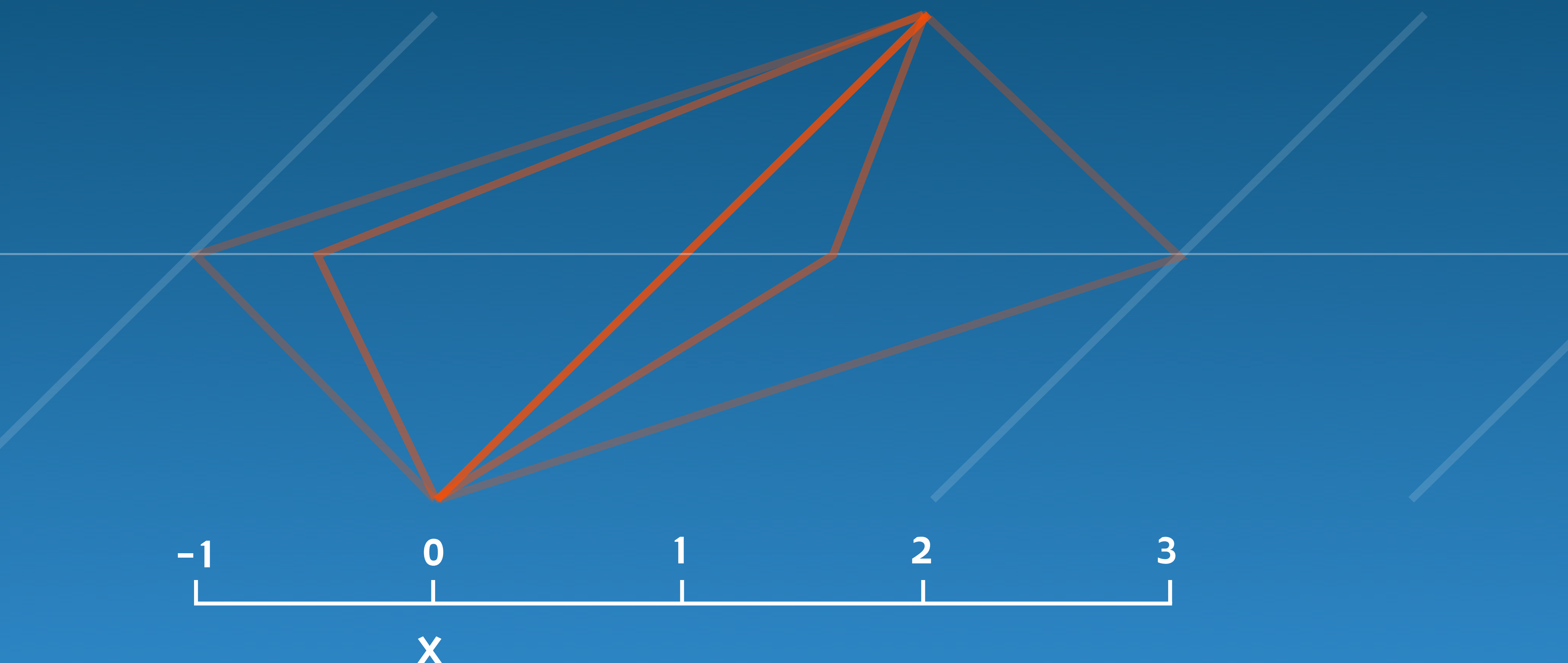
North light scheme, typical gradient of illumination

Example III



North light scheme, gradient of illumination

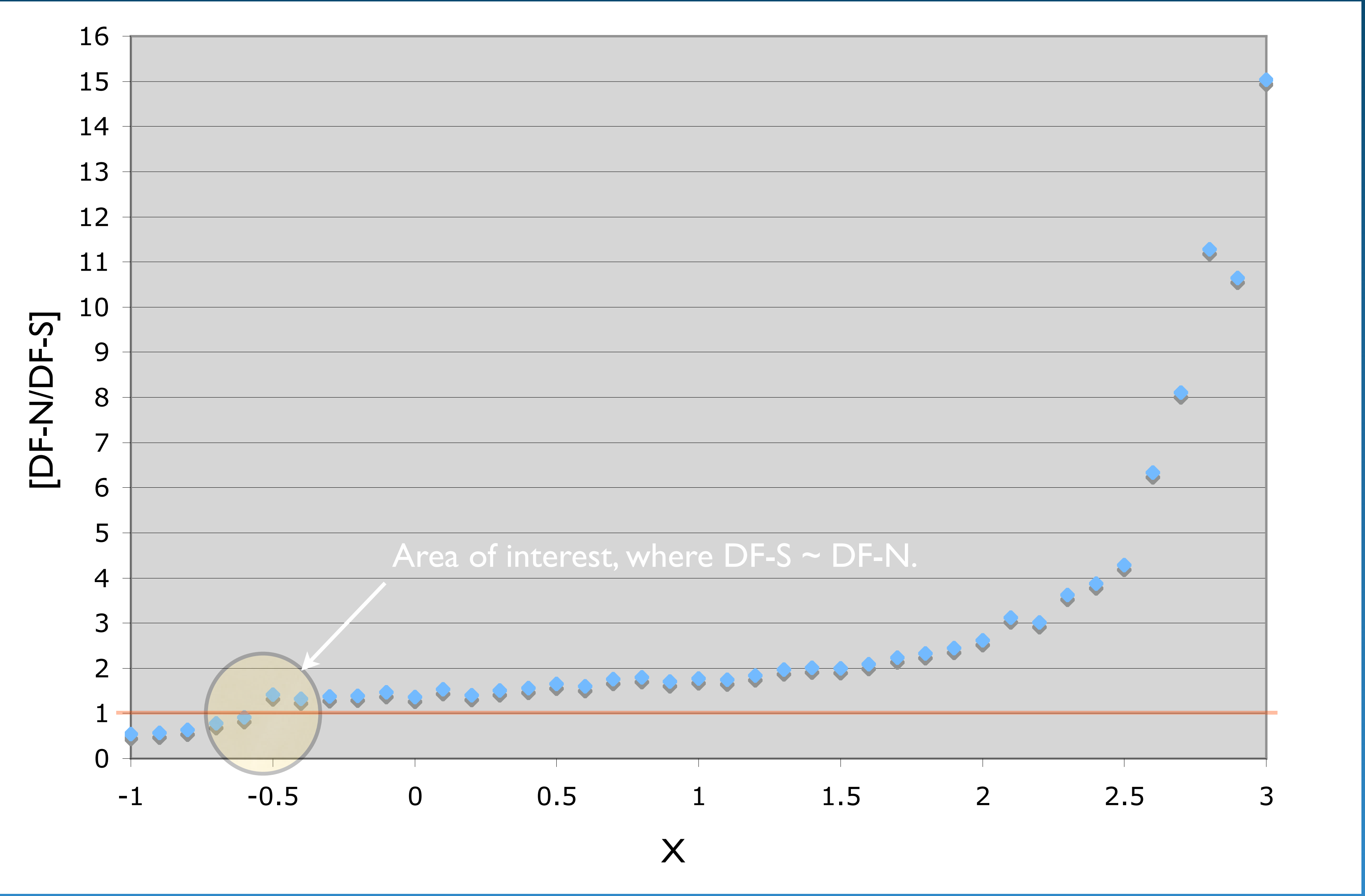
Example III



possible configurations of louvre profile

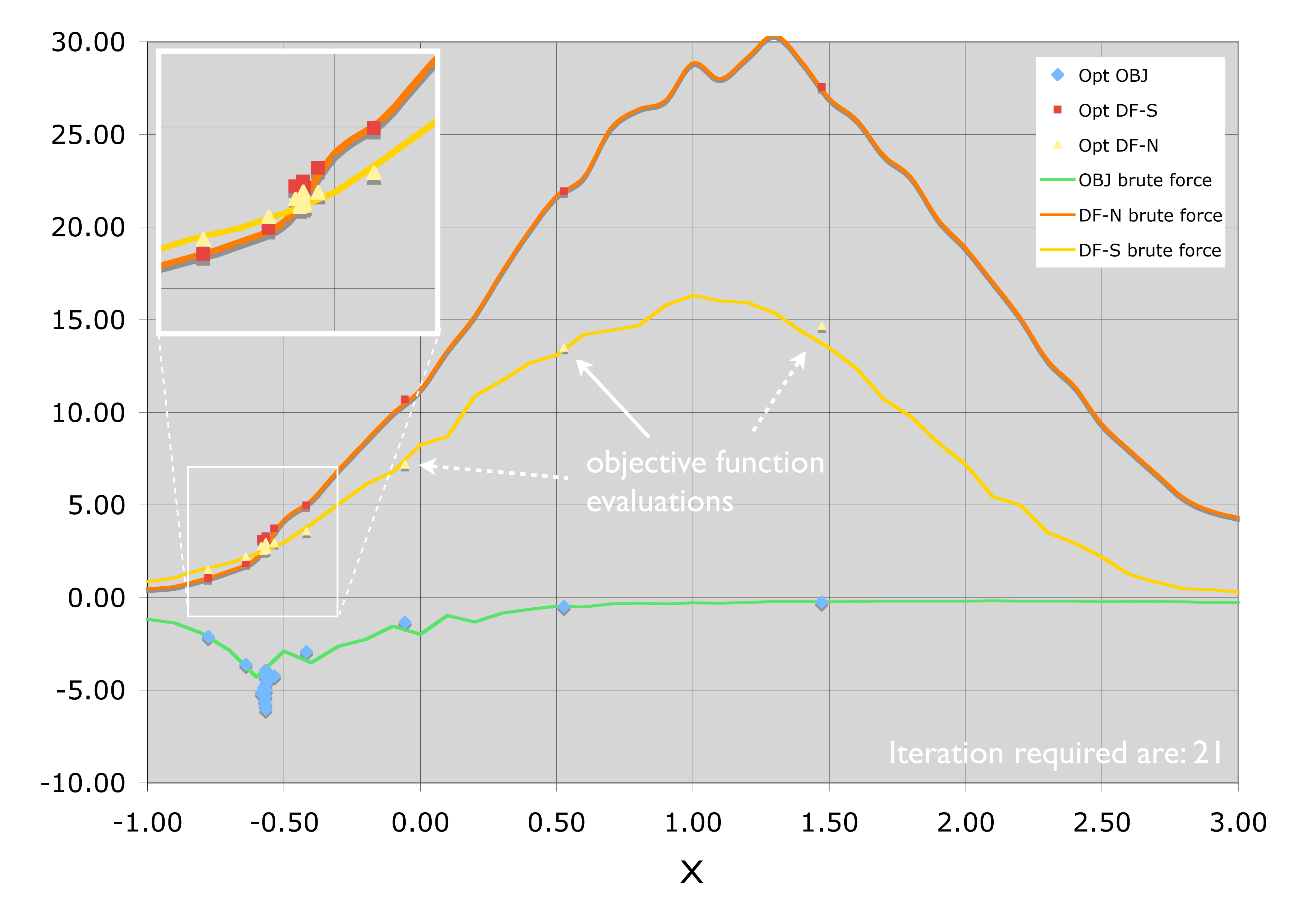
ArupLighting

Example III



Plot of f(x)=(DF-N/DF-S) for x in [1,3].

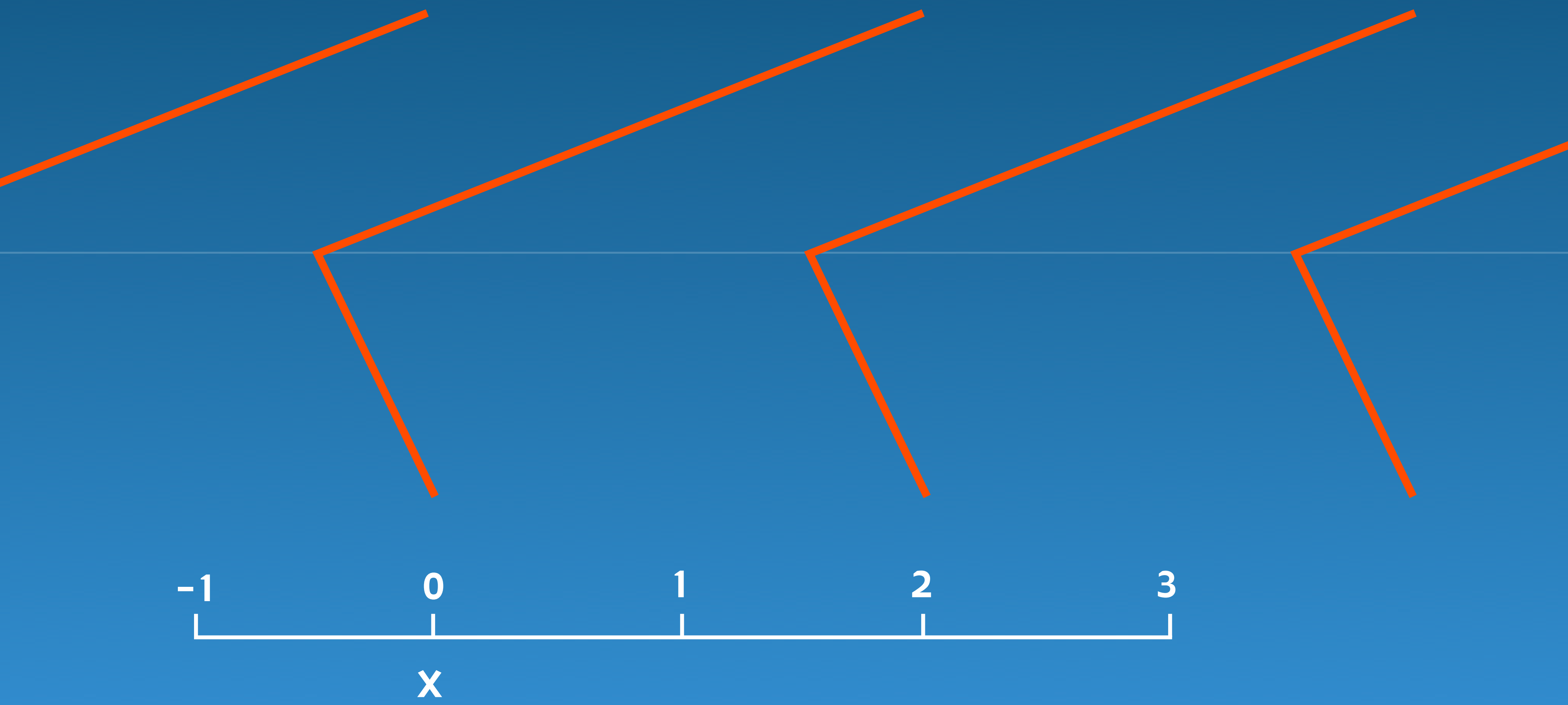
Example III



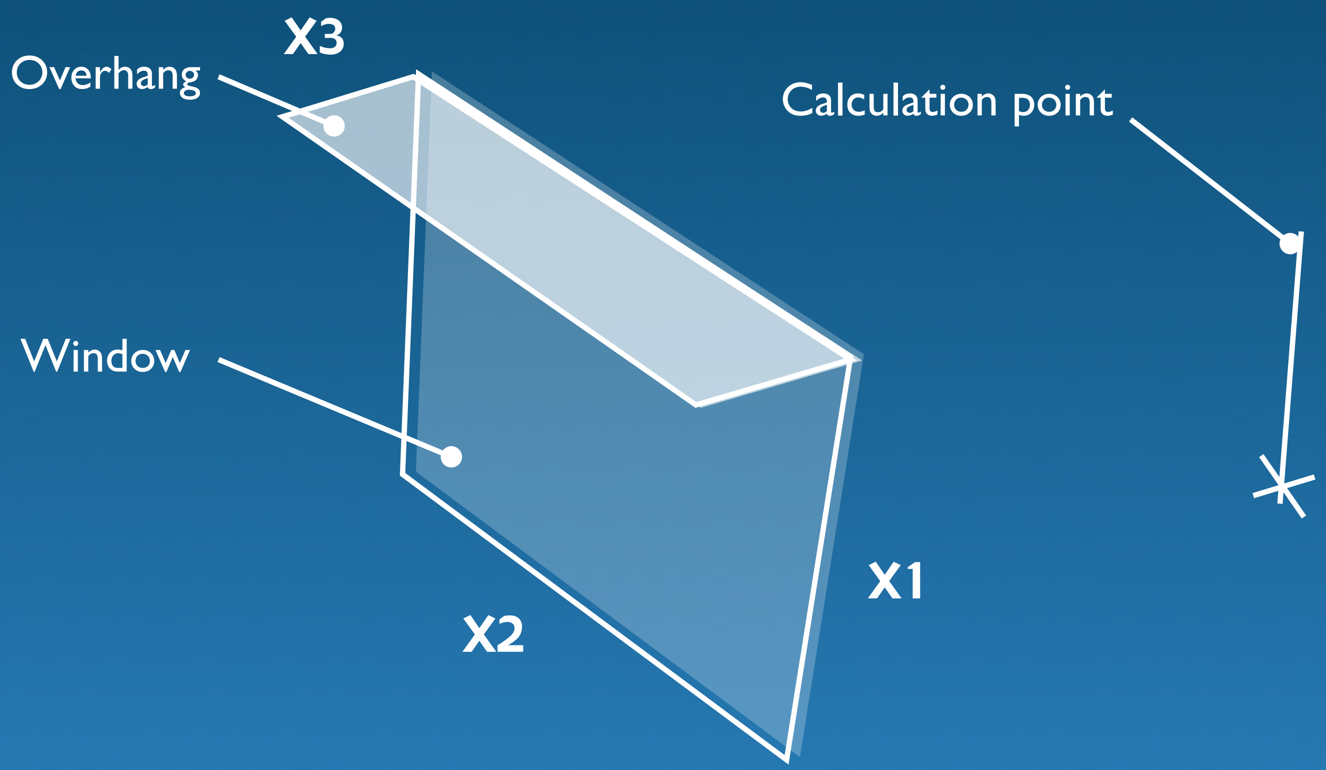
30.00
25.00
20.00
15.00
10.00
5.00
0.00
-5.00
-10.00
-1.00
-0.50
0.00
0.50
1.00
1.50
2.00
2.50
3.00
X
Opt OBJ
Opt DF-S
Opt DF-N
OBJ brute force
DF-N brute force
DF-S brute force
objective function evaluations
Iteration required are: 21

Example III

Optimised louvre profile:
Iterations: 21
x = -0.57
DF-N = 3%
DF-S = 2.8%



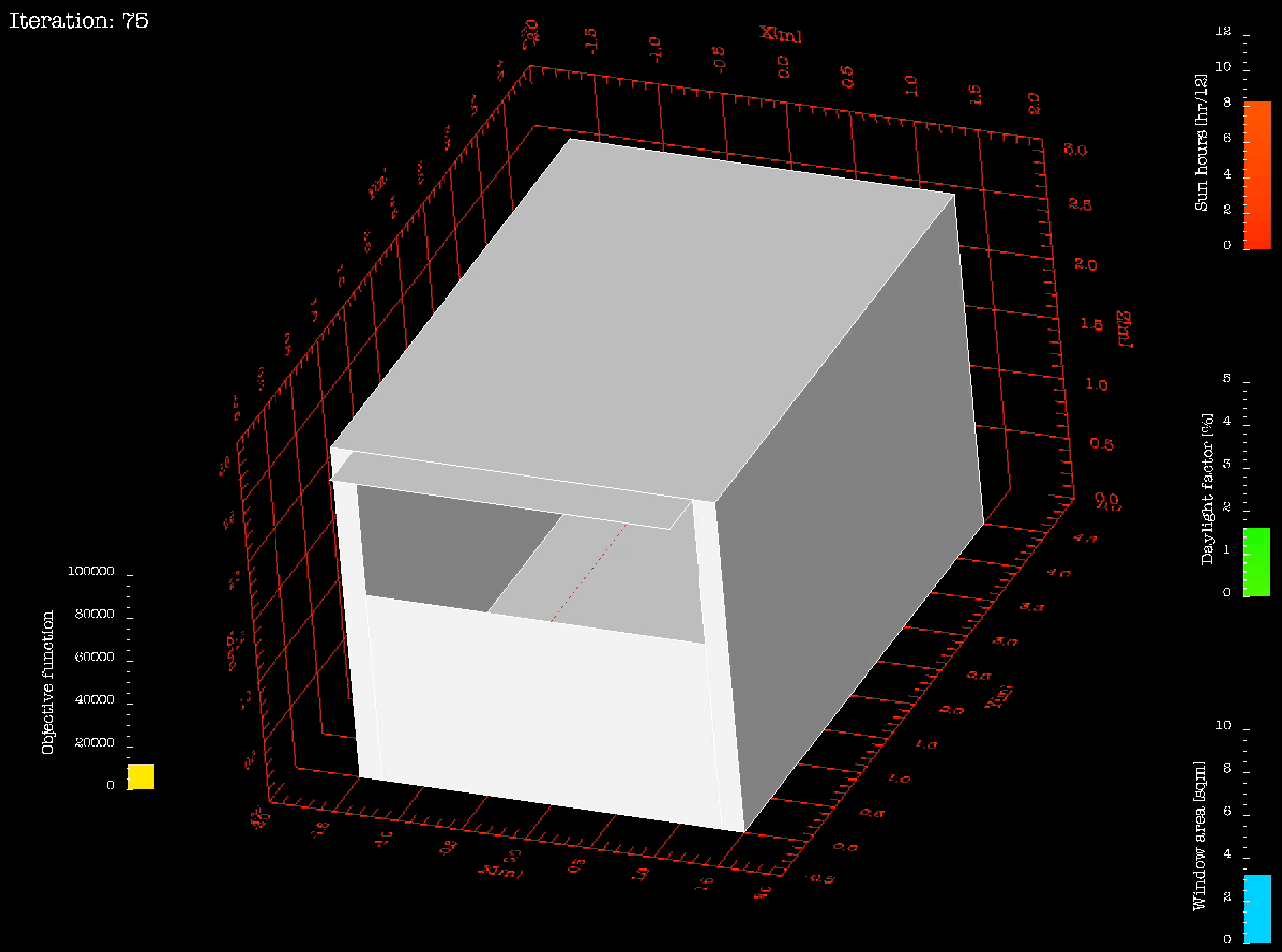
Example IV



Optimisation engine: simplex search

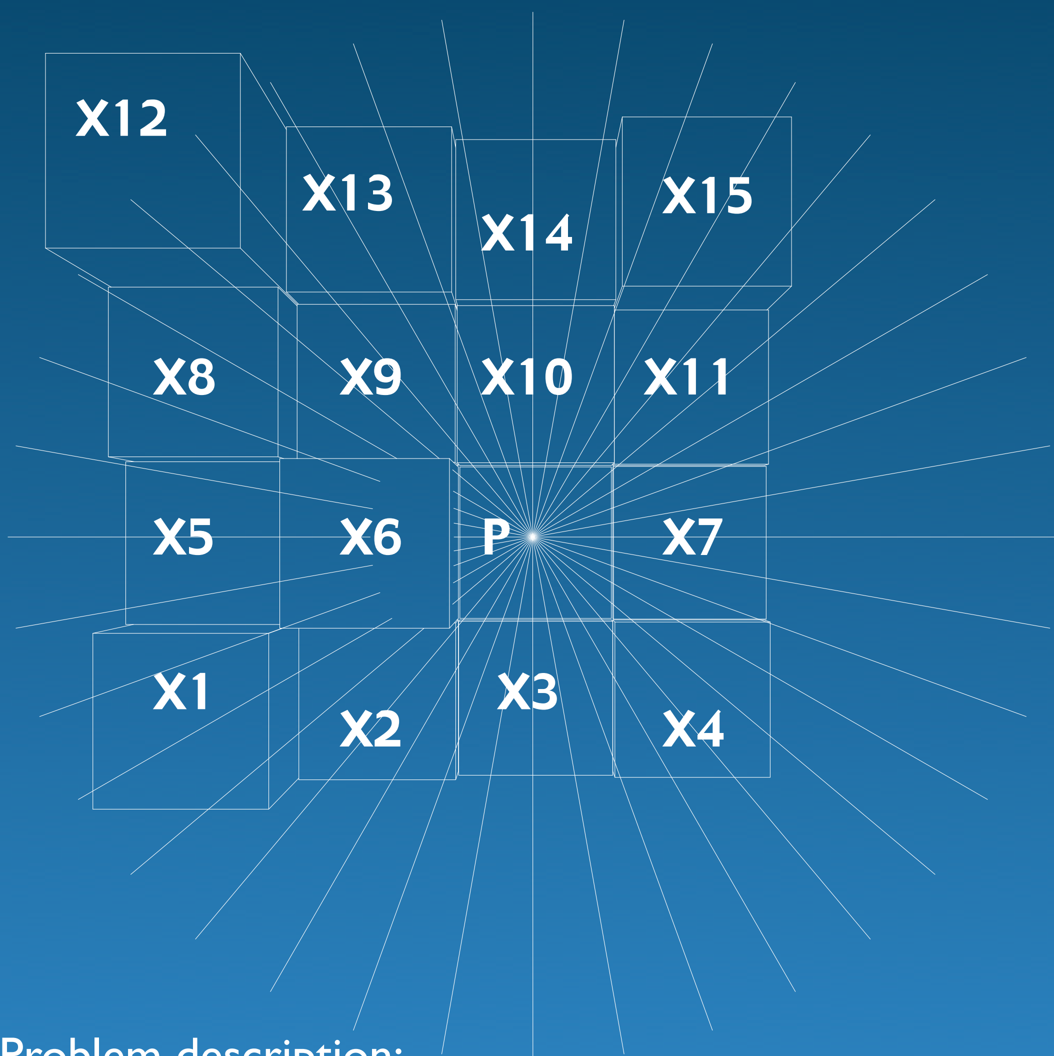
Problem description:
window shape and overhang are generated parametrically: there are 3 variables.
Daylight factor is maximised while area of window and sun hours are minimised.

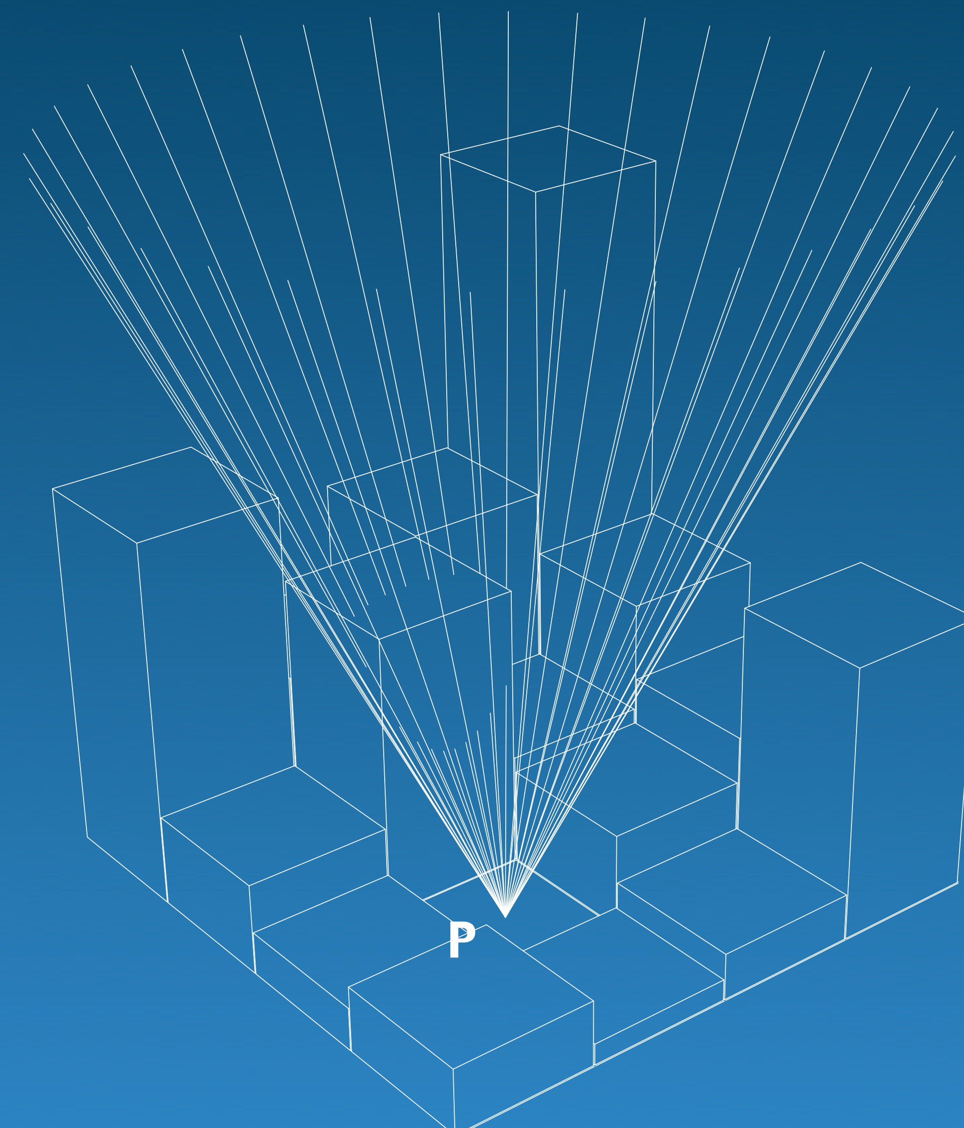
Example IV



Iteration: 75
X[m]
Y[m]
Z[m]
Sun hours [hr/12]
Daylight factor [%]
Window area [sqm]
Objective function

Example V - part 1

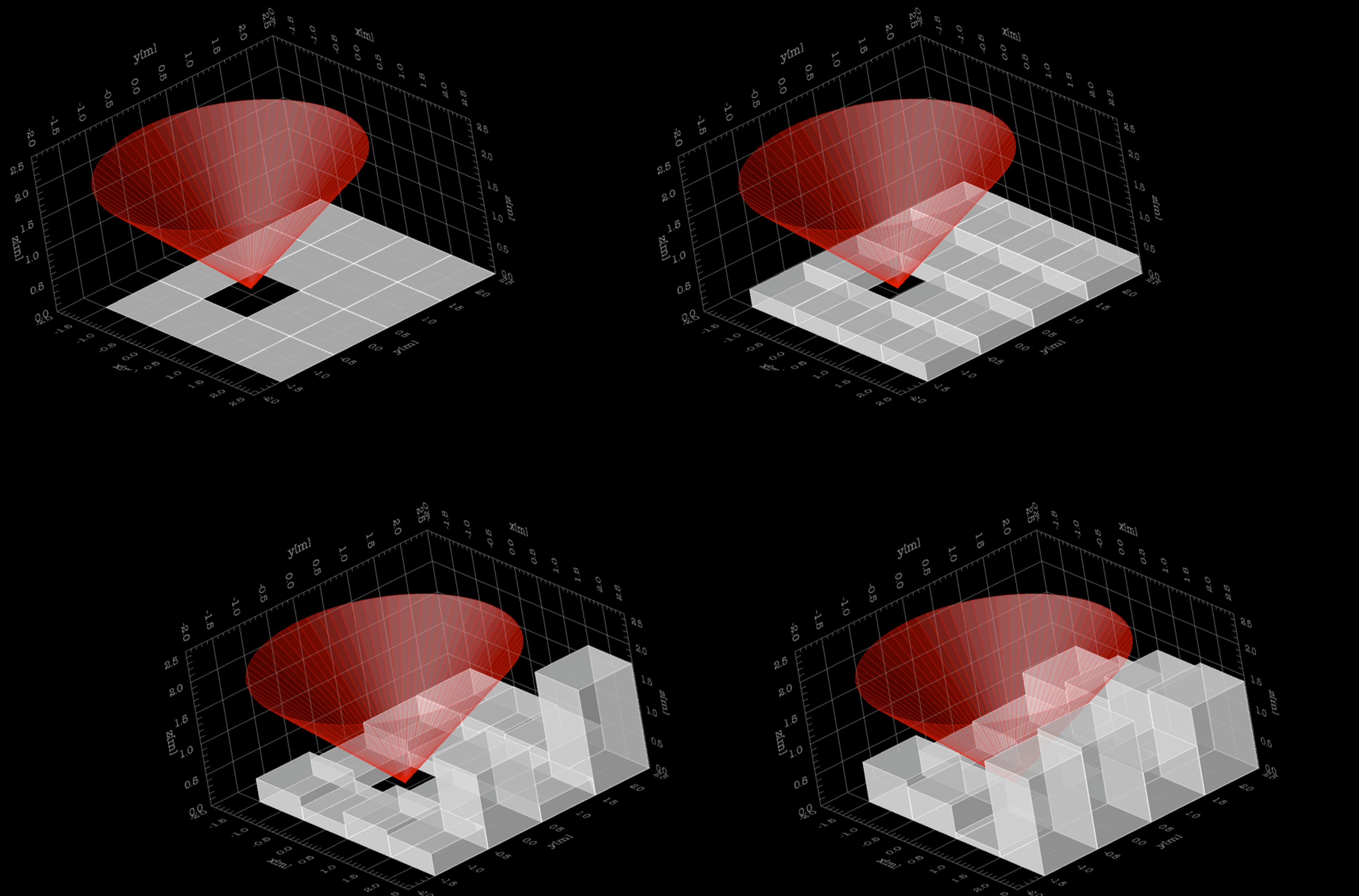




Problem description:
15 solid blocks have variable heights.
There are 15 variables in total.
Sky visibility is tested for a point between the blocks.
Volume is maximised without obstructing any of the viewing directions.

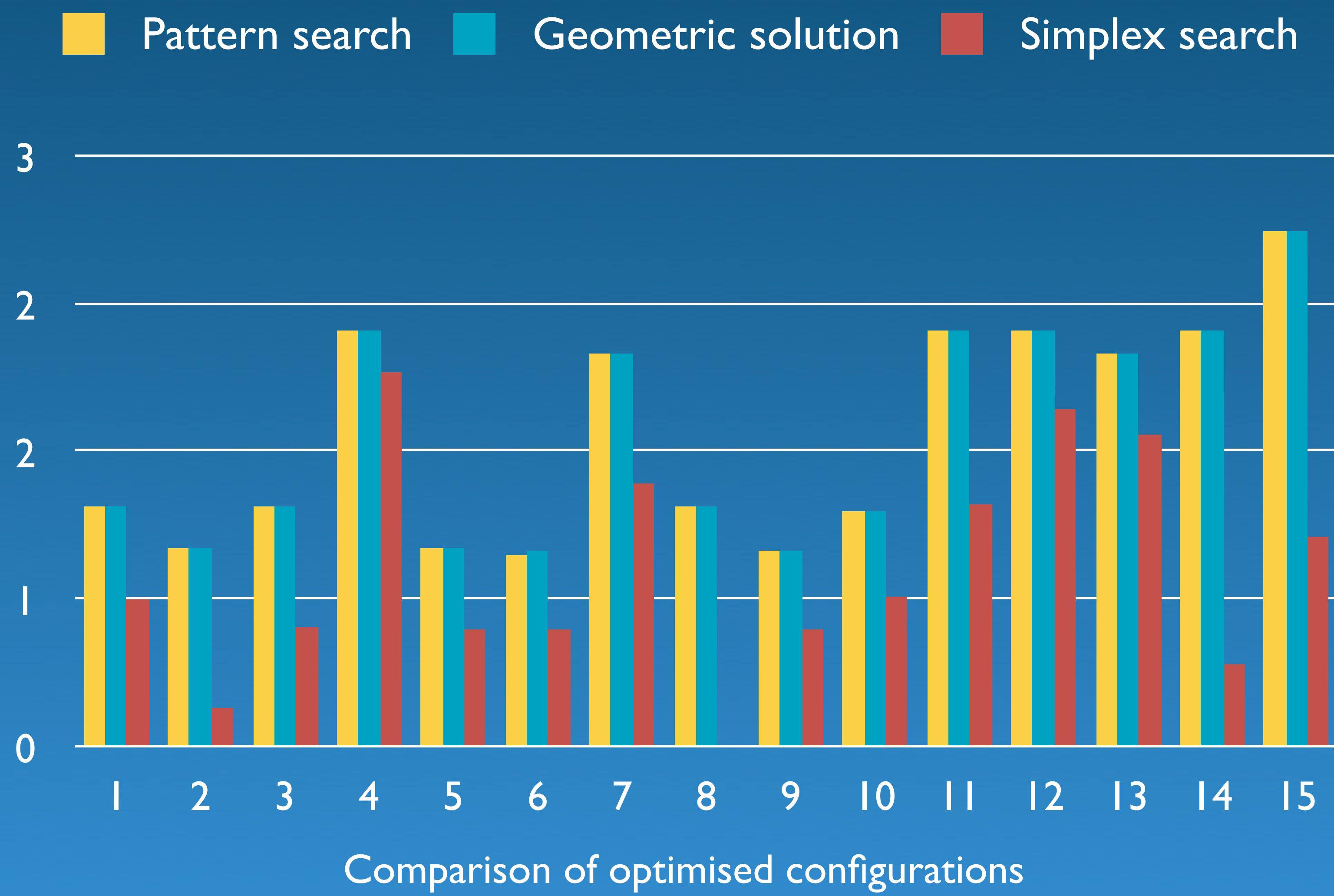
Optimisation engine:
simplex search, pattern search and analytic solution.

Example V

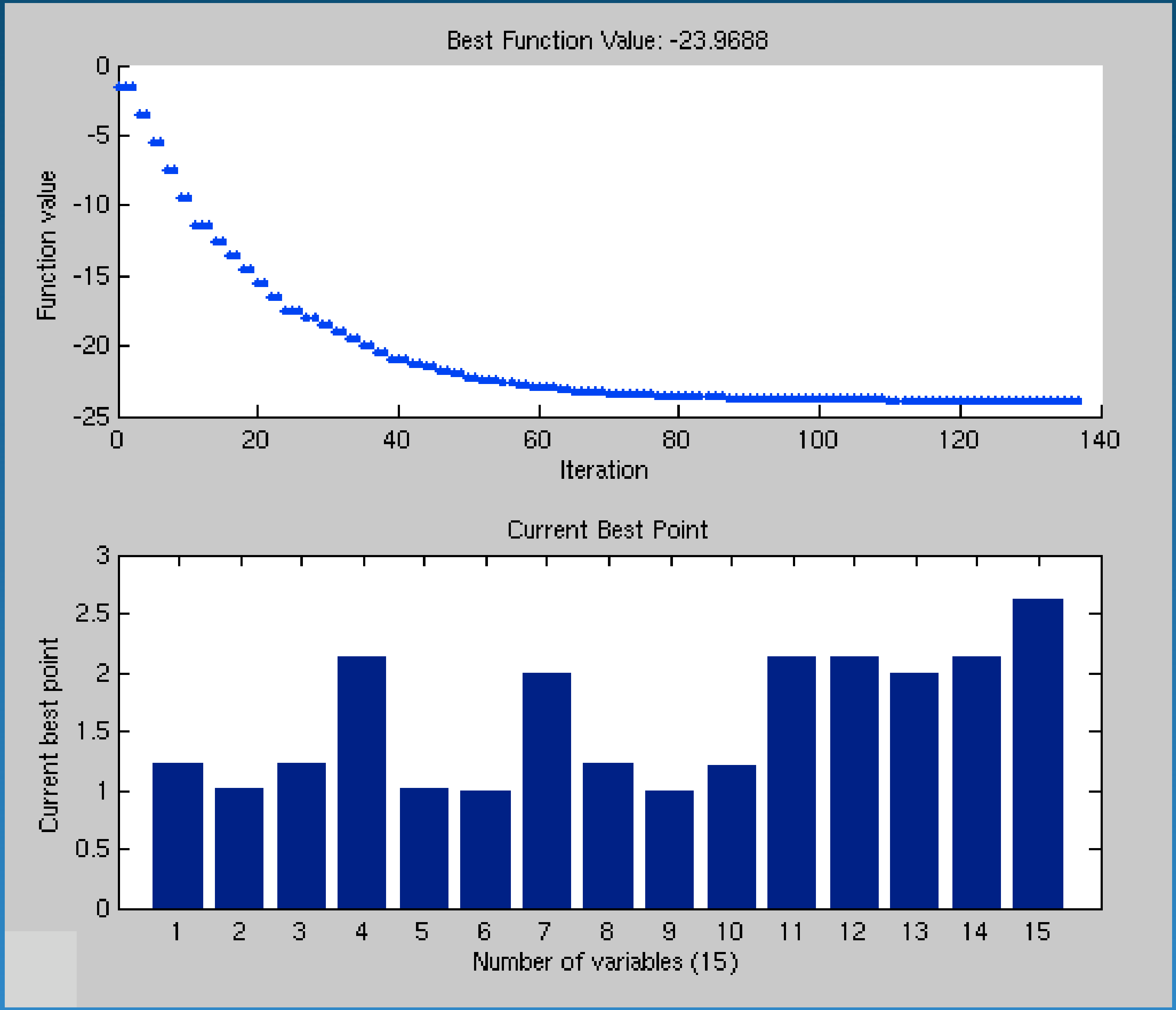


The problem can be easily solved geometrically.

Example V - part 1



Example V - part 1



Matlab, Pattern search toolbox screenshot of results

Example V - part 1

Summary:

- ☞ Pattern search engine find a configuration that is virtually coincident with the analytic solution.
- ☞ This lead us to investigate further application of this jewel...

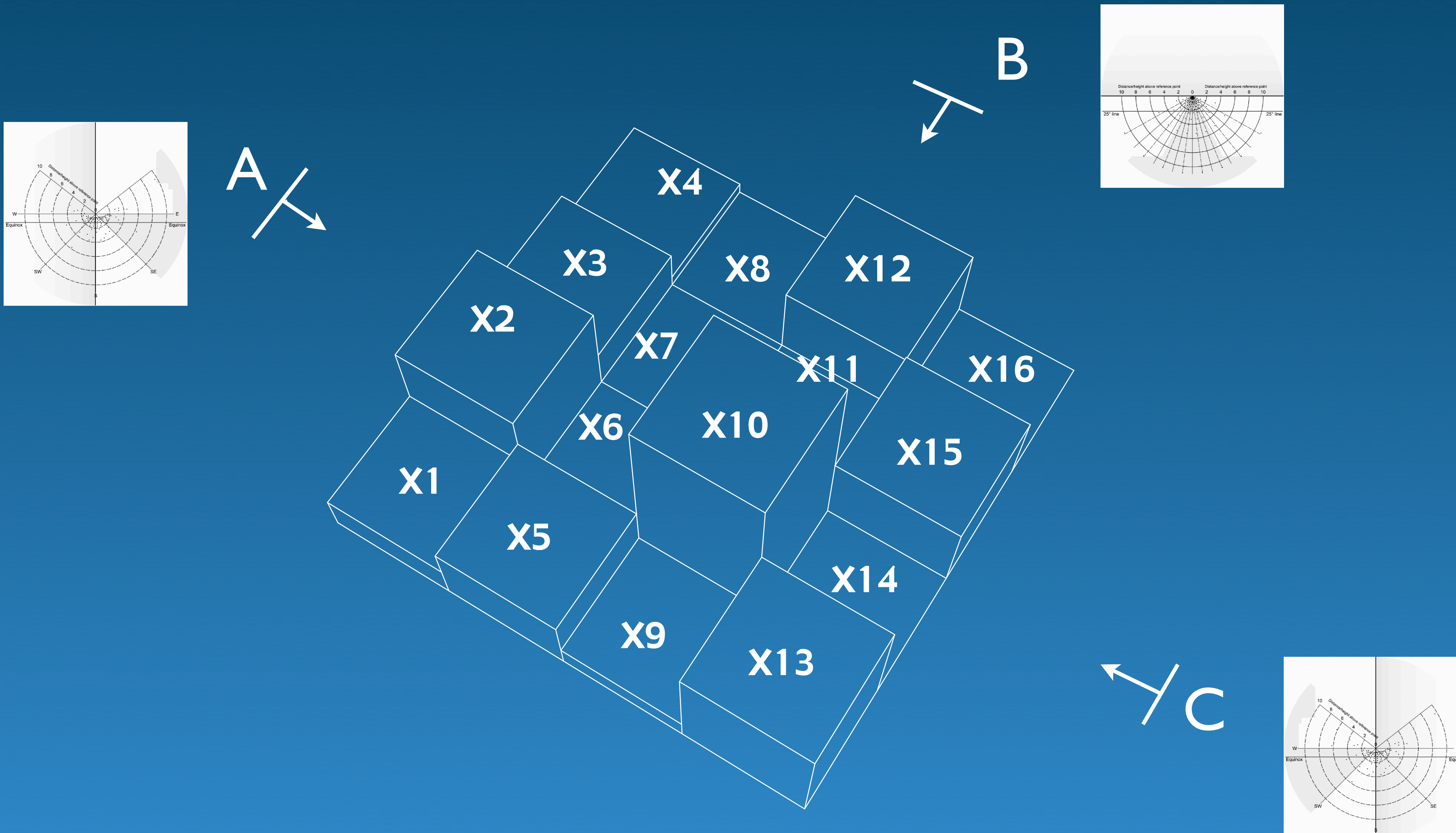
Example V - part II

Problem:

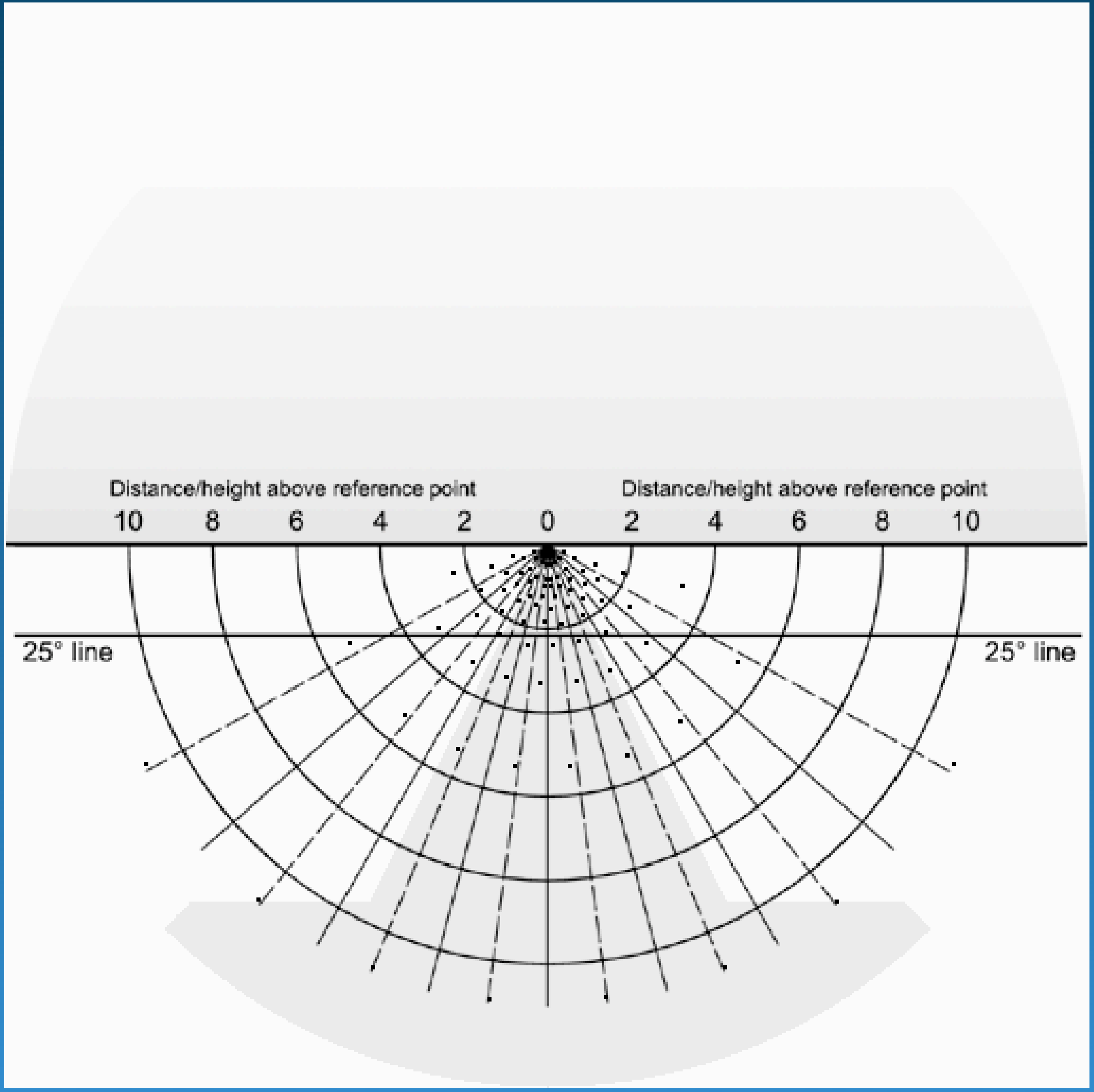
maximise the volume of a development whilst achieving at least the minimum required skylight availability score for a series of test points:

- SA>27 for all points
- Optimise volume

Example V - part II



Example V - part II



Example V - part II

Alternative approaches:

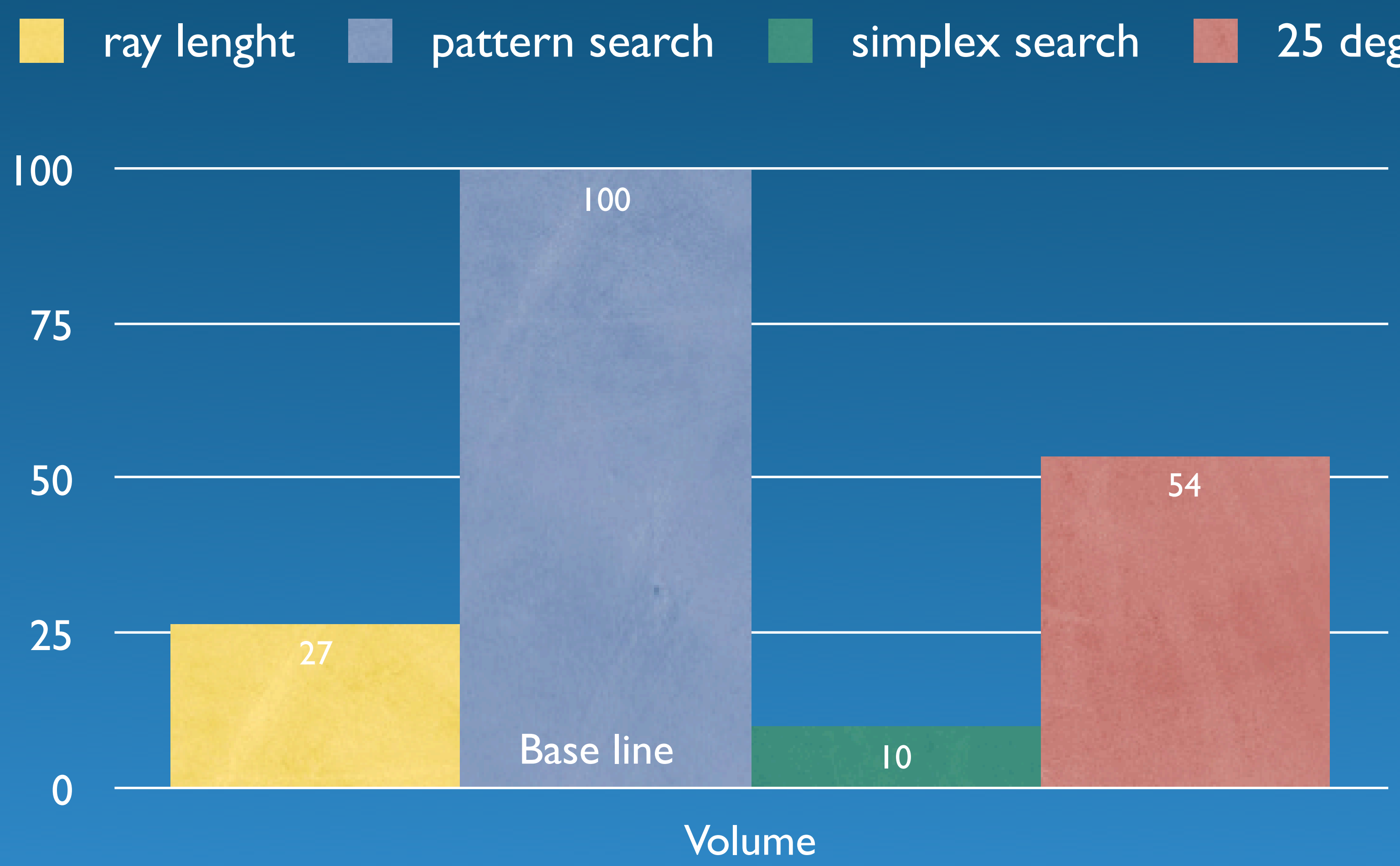
I. Radiance + Octave (GA, simplex)

II. Radiance with “ray length trick” (geometric shortcut).

III. Radiance with Matlab and pattern search.

IV. 25° line rule

Example V - part II



Comparison of optimised configurations

Example V - part II

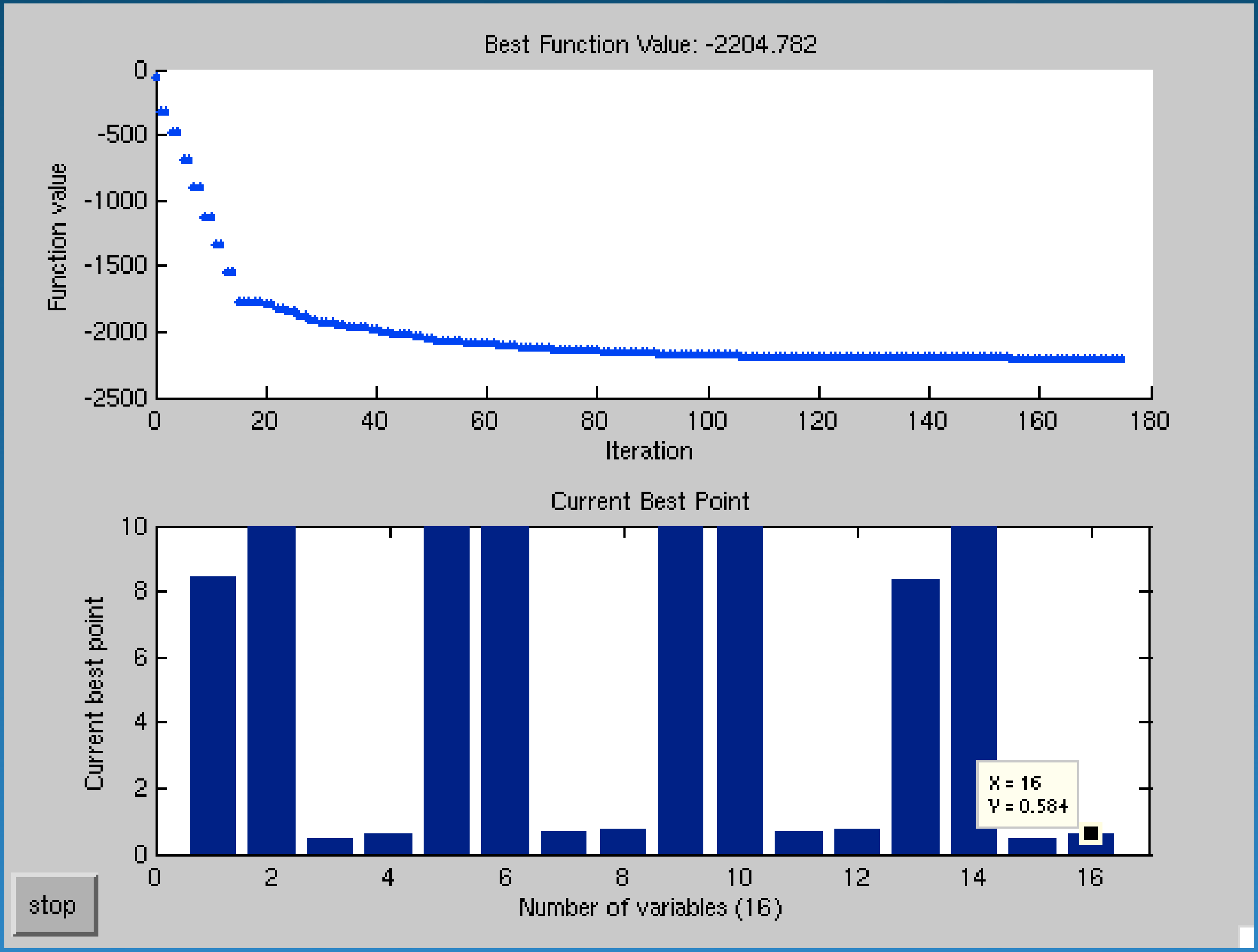
Simplex doesn't really work because of the difficulty to write a decent objective function, not too spiky.

'Ray length' approach produces solution that are not optimal and somehow less usable. It has been a promising idea but it has been abandoned because...

☞ Matlab pattern search quickly converges to the maximum solution. Even with a really spiky objective function!

the 25° line rule works by definition, but the volume is less than the Matlab solution.

Example V - part II



Matlab pattern search screenshot, there are 16 variables

Example V - part III

By twisting the problem is possible to use another approach:

I. check volume, if V>T register configuration into set S

II. calculate the performance of all the configuration in S

III. order them by volume or performance

The volume test can easy be converted in a test over the measure of floor area... the approach is really flexible about this.

Example V - part III

This approach has the great advantage to find all the configurations that are sustainable both economically and environmentally but requires a relevant computational effort.

Example V - part III

For example:

10 buildings development,

each building has 10 floors

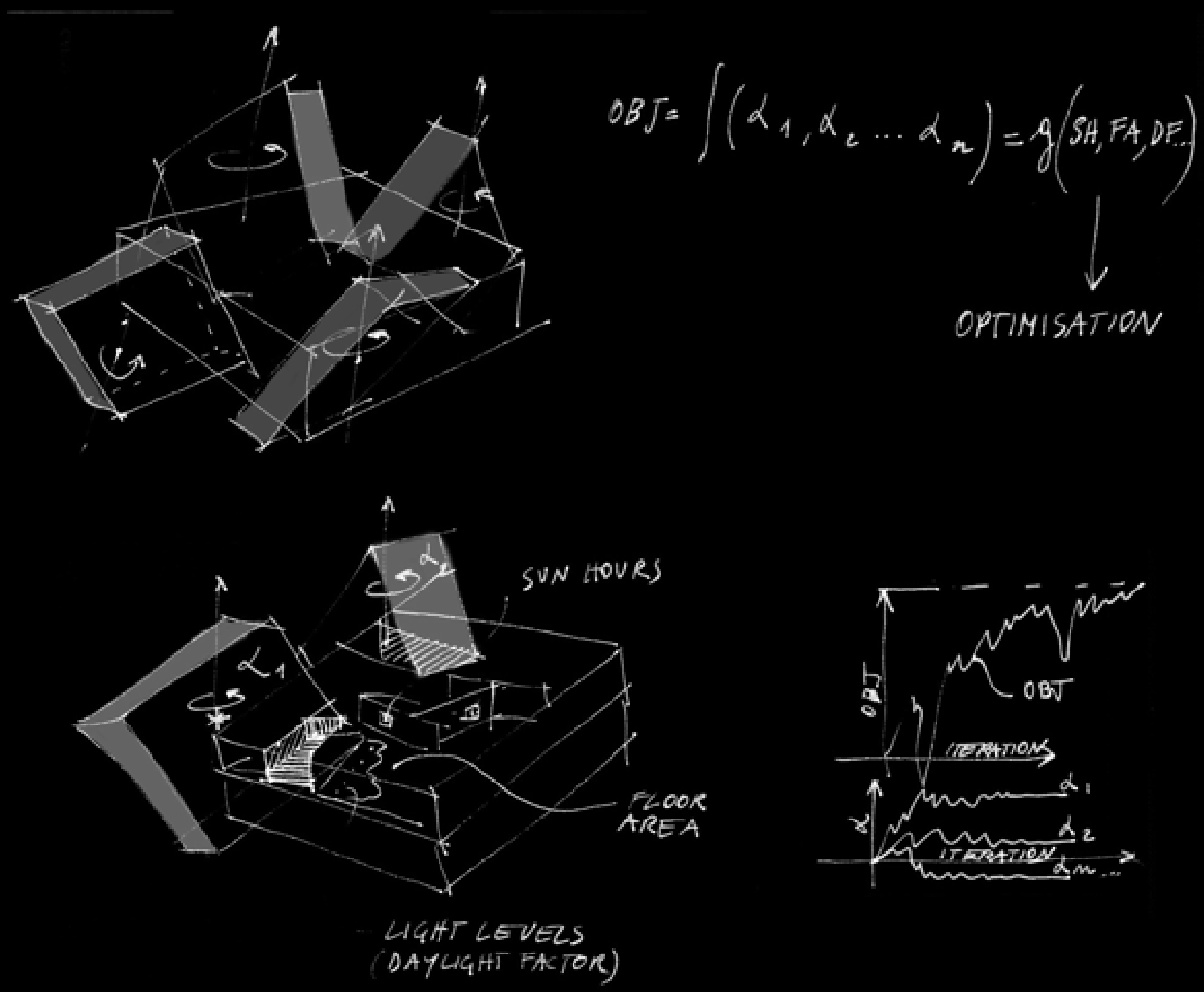
gives a total number of possible configurations: N = 10^10.

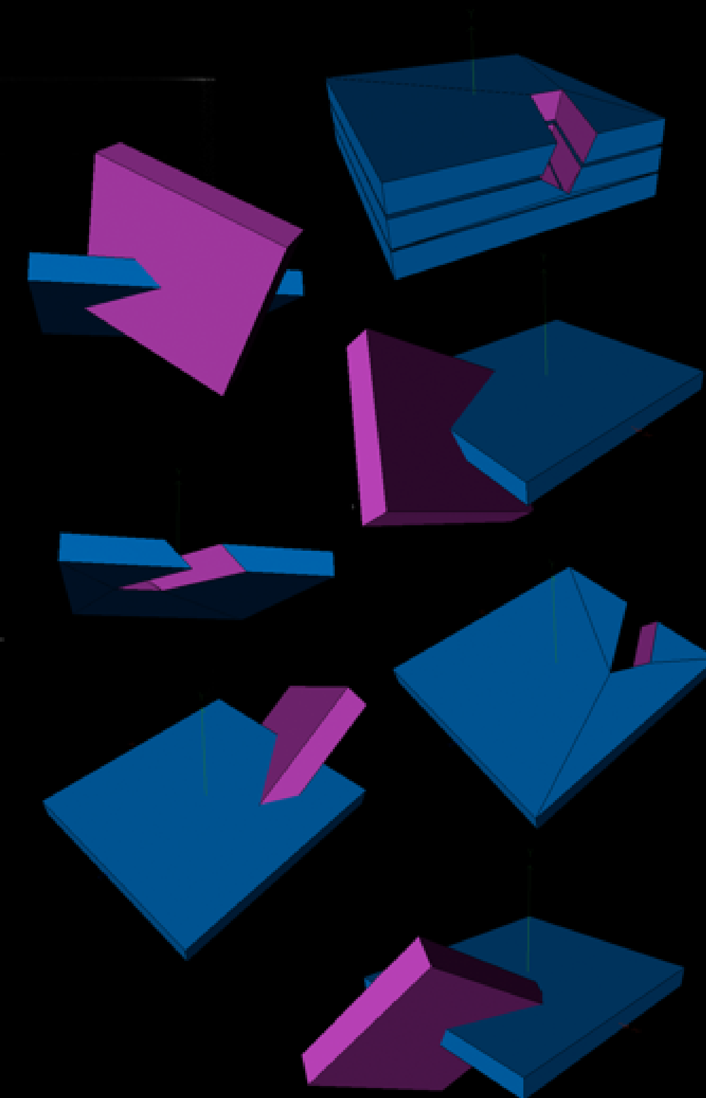
If we are interested in just the solution that have at least 90% of the maximum floor area the number of configurations reduces dramatically to Nr ~ 10^5

By parallelising the calculation it is possible to evaluate all these configurations in a reasonable time.

Case studies

Estonian museum of Art

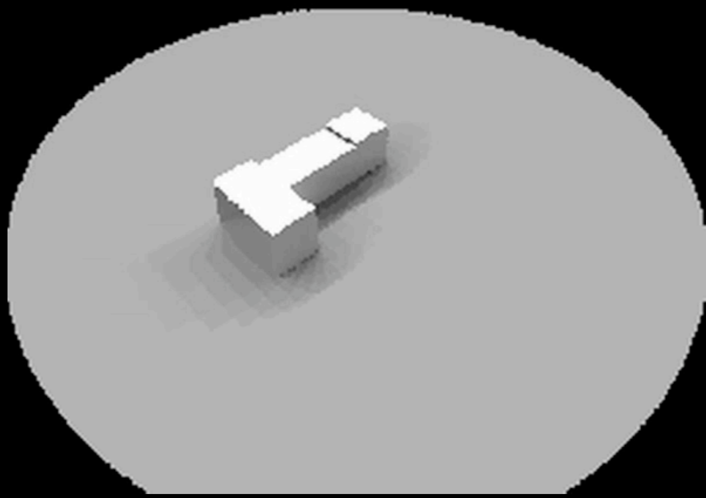


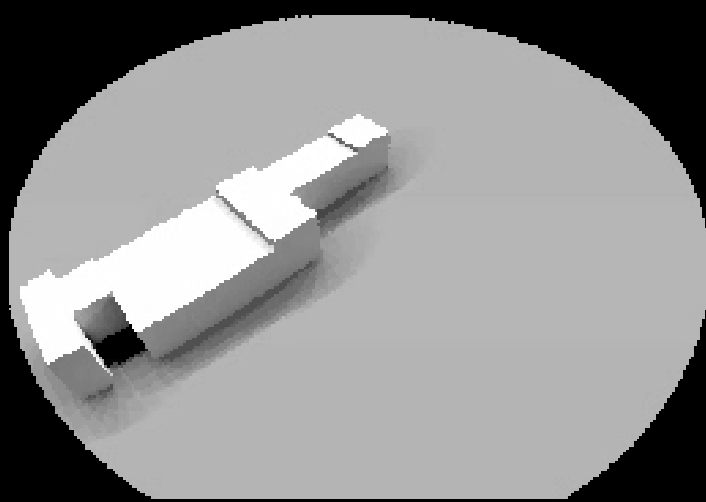


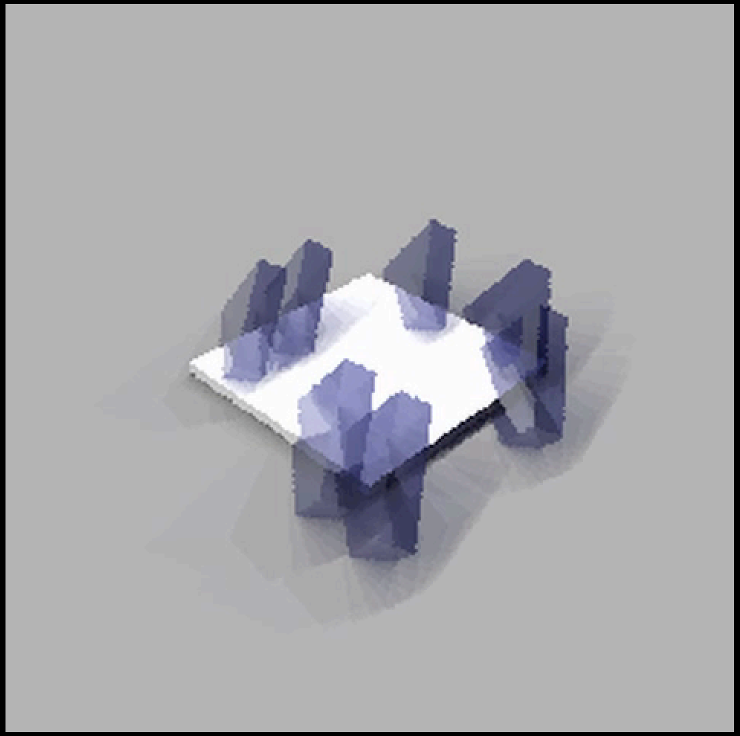
Optimisation algorithm development - concept
Estonian museum of Art
Competition entry with Gianni Botsford Architects

ArupLighting

Estonian museum of Art

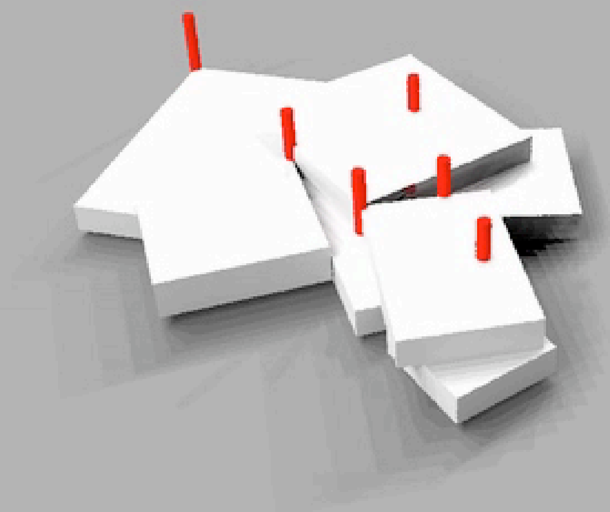






Optimisation algorithm development
Estonian museum of Art
Competition entry with Gianni Botsford Architects

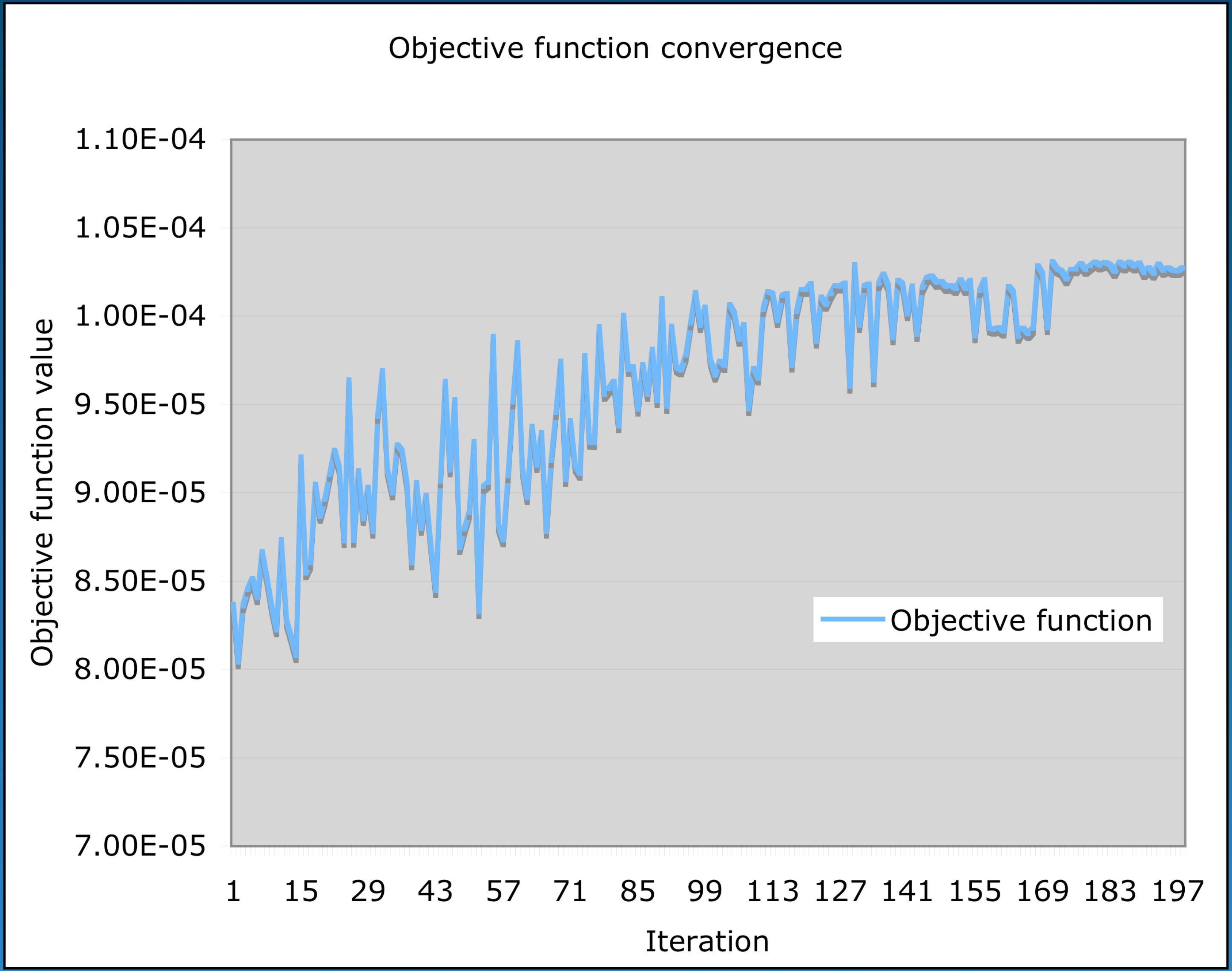
Estonian museum of Art



Estonian museum of Art massing - maximum daylight factor and minimum sun hours.
Competition entry with Gianni Botsford Architects

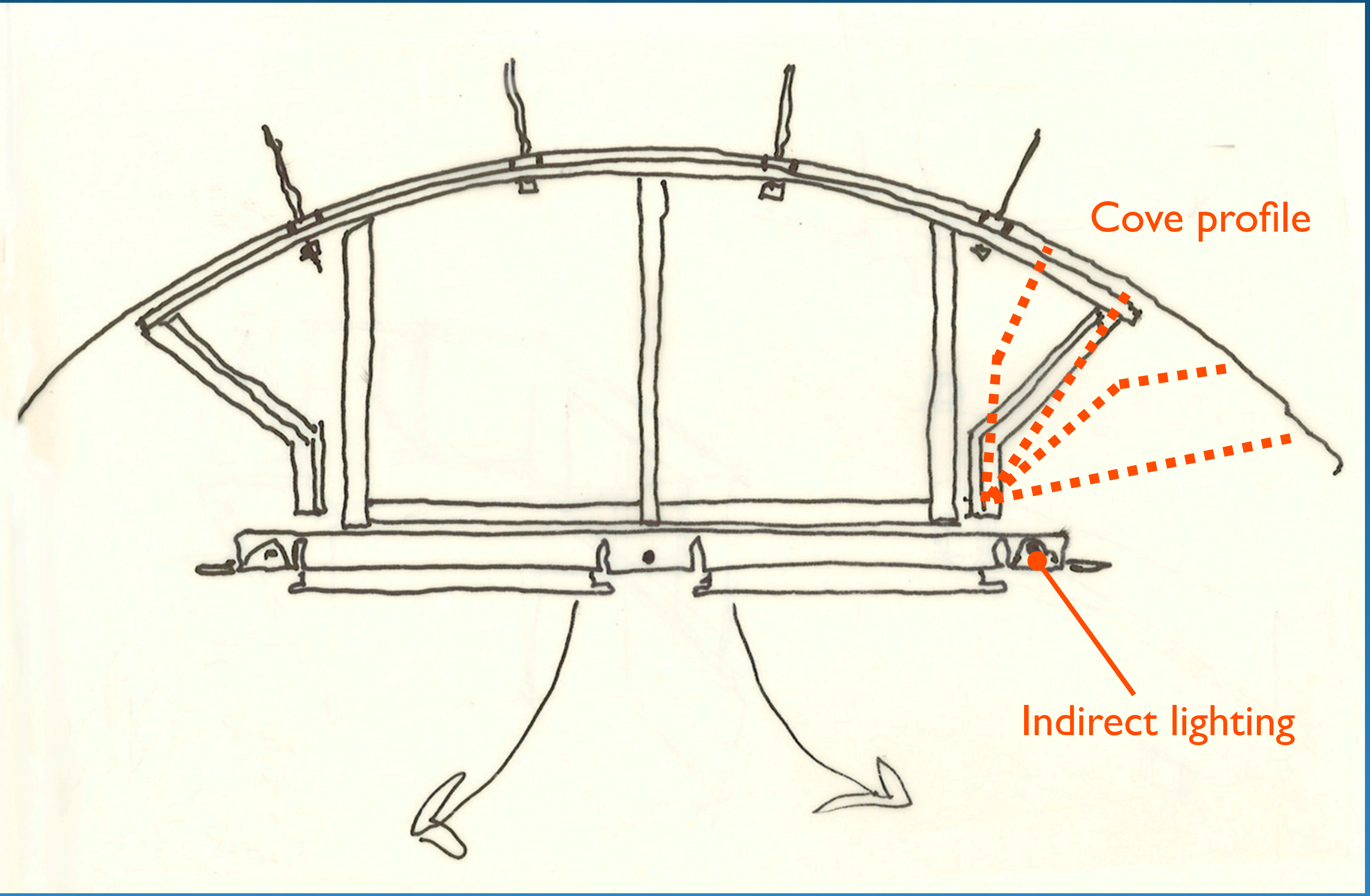
ArupLighting

Estonian museum of Art



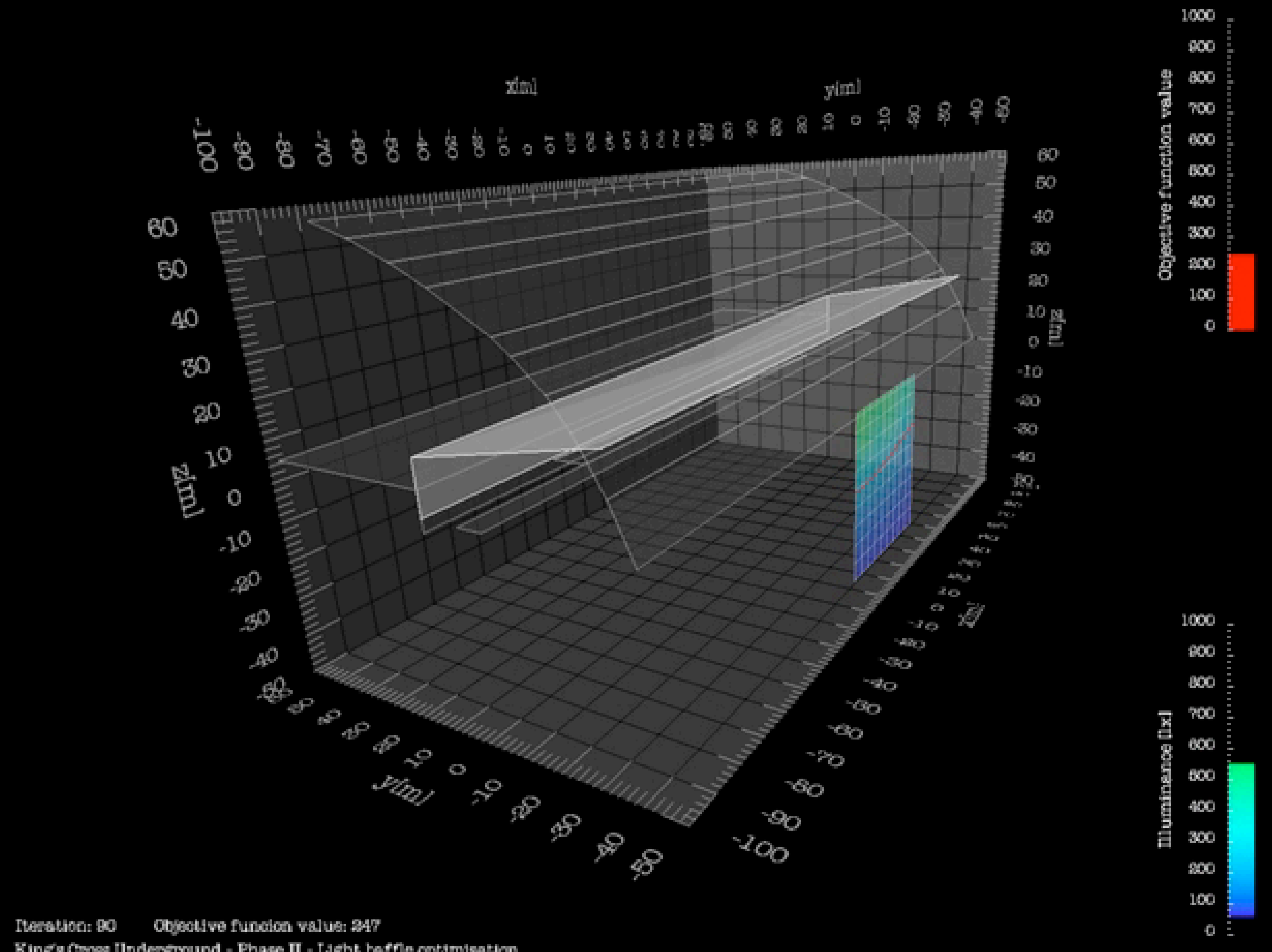
Convergence vs. objective function value, simplex search.
Competition entry with Gianni Botsford Architects

King's Cross station cove lighting



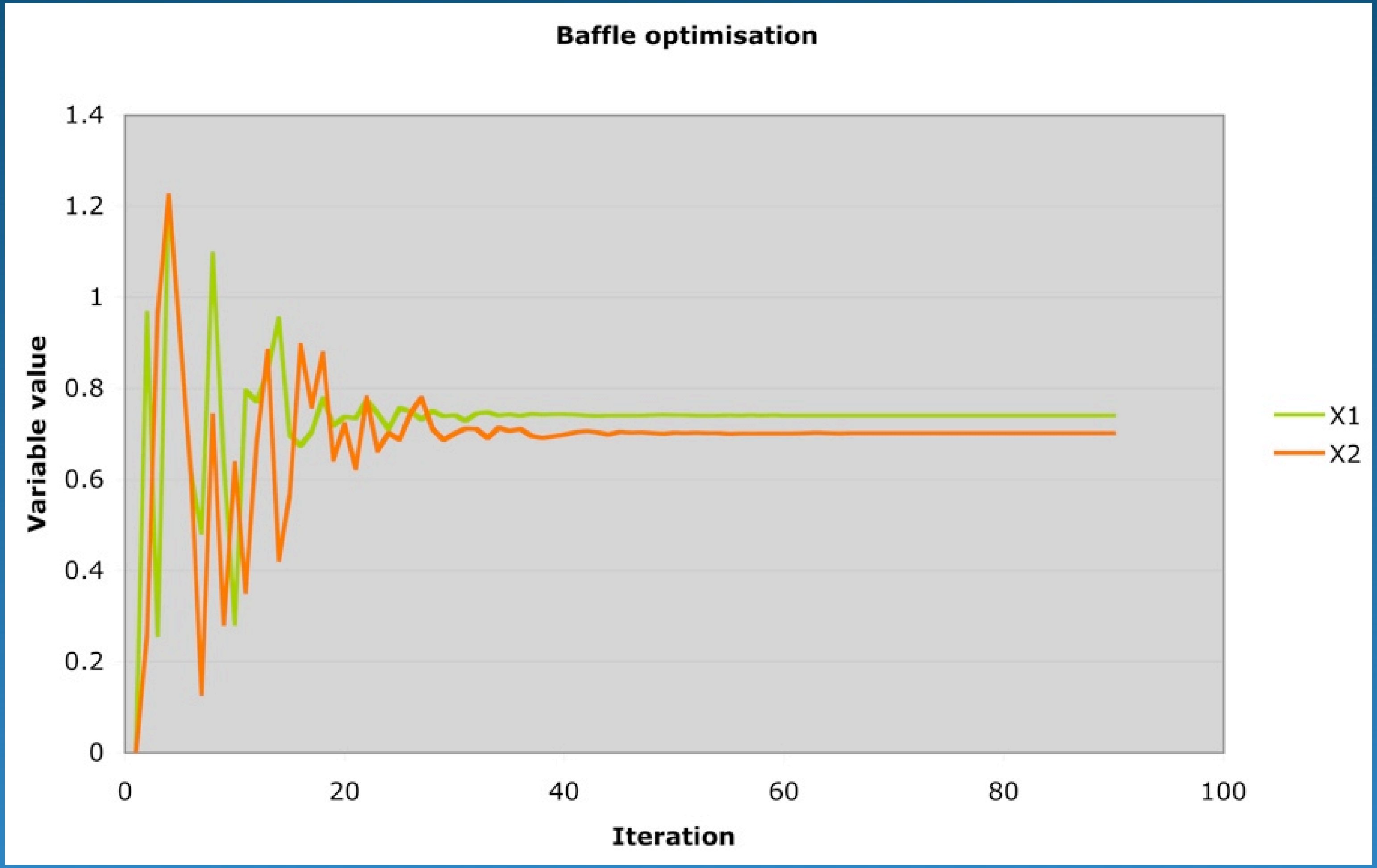
Initial concept from the architect. The idea is to maximise the indirect lighting of the side wall by reshaping the cove profile.

King's Cross station cove lighting



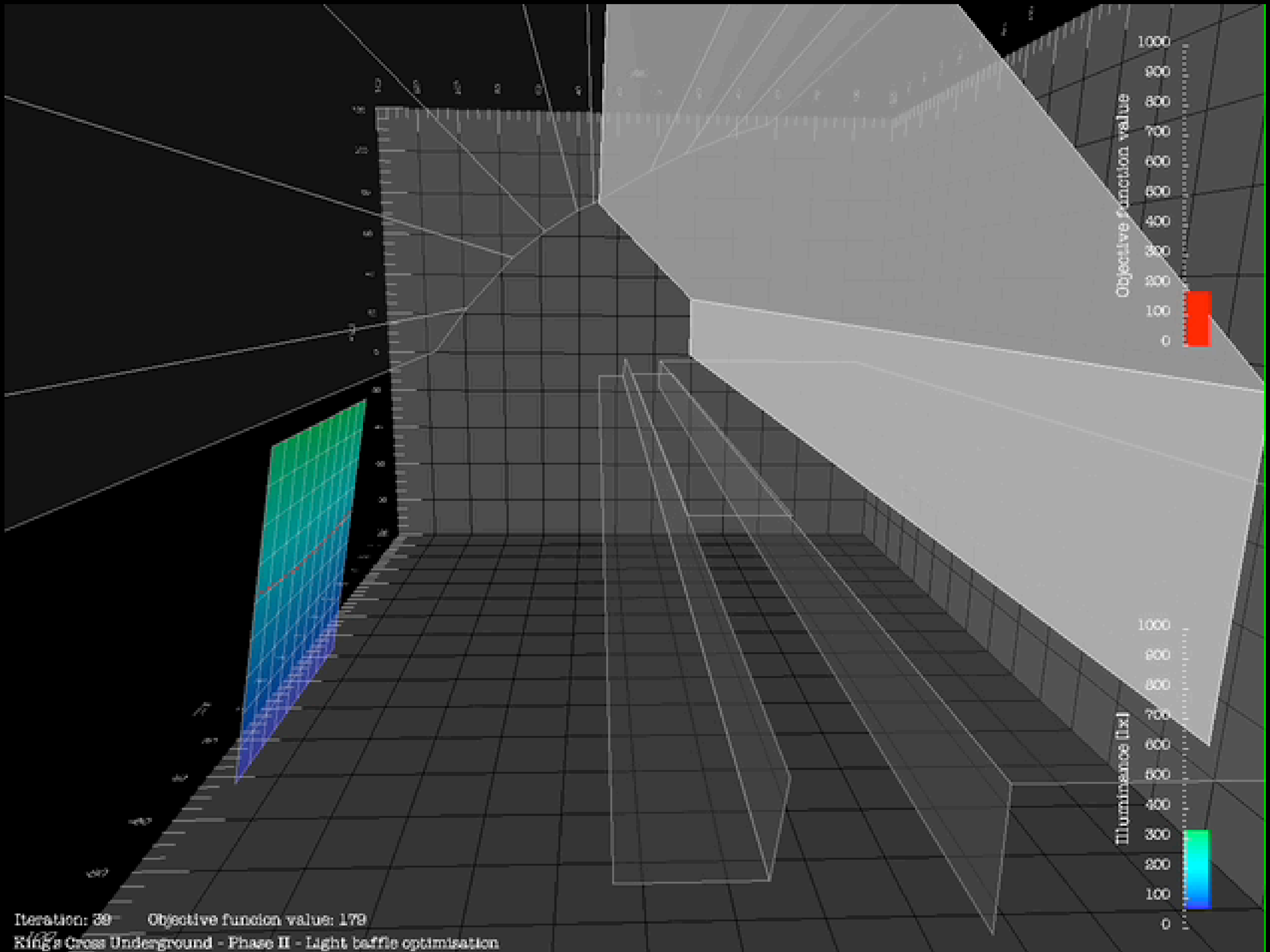
ArupLighting Optimisation of the profile with two variables. Objective function is Illuminance on wall.

King's Cross station cove lighting



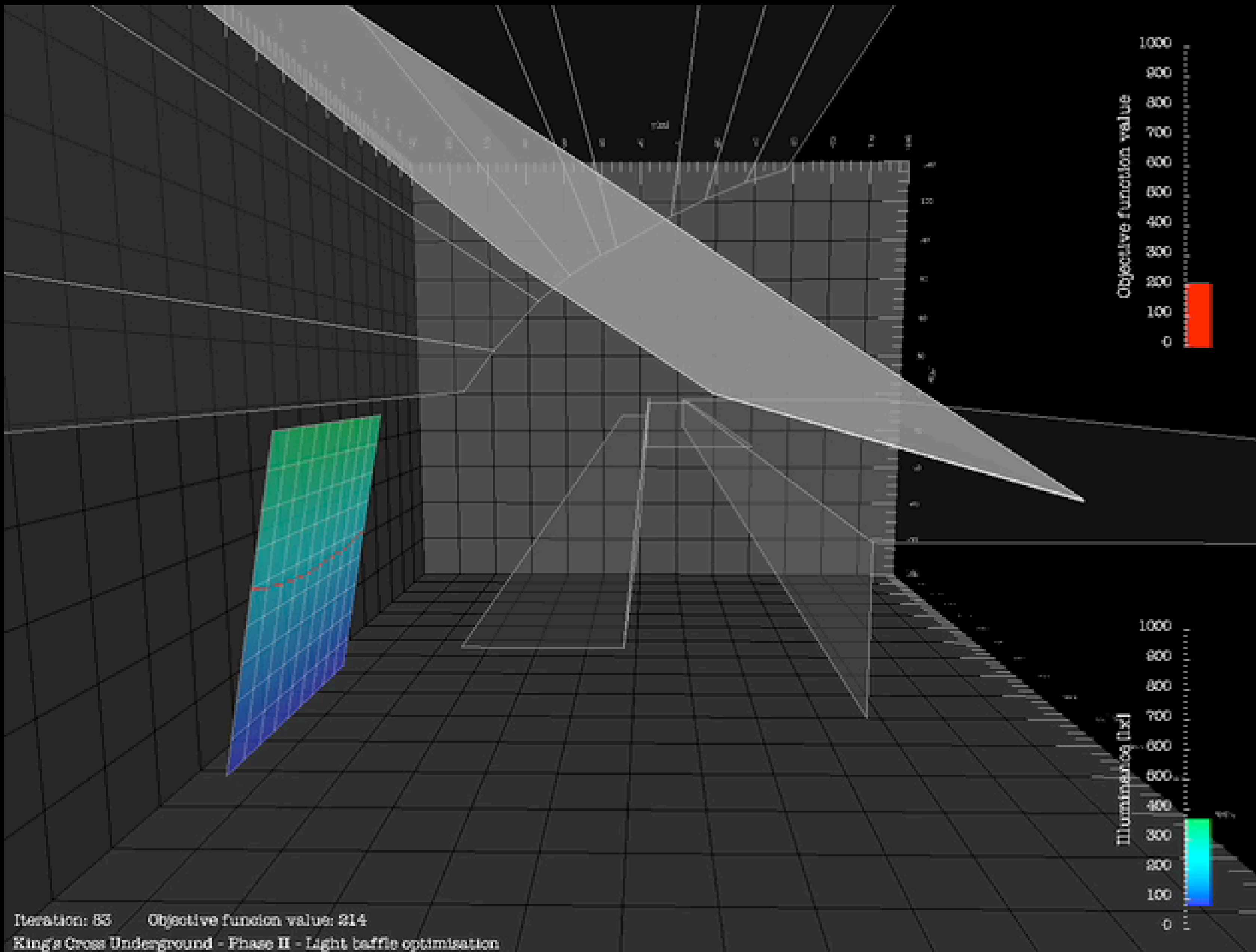
Convergence of solution vs. iteration, simplex search.

King’s Cross station cove lighting



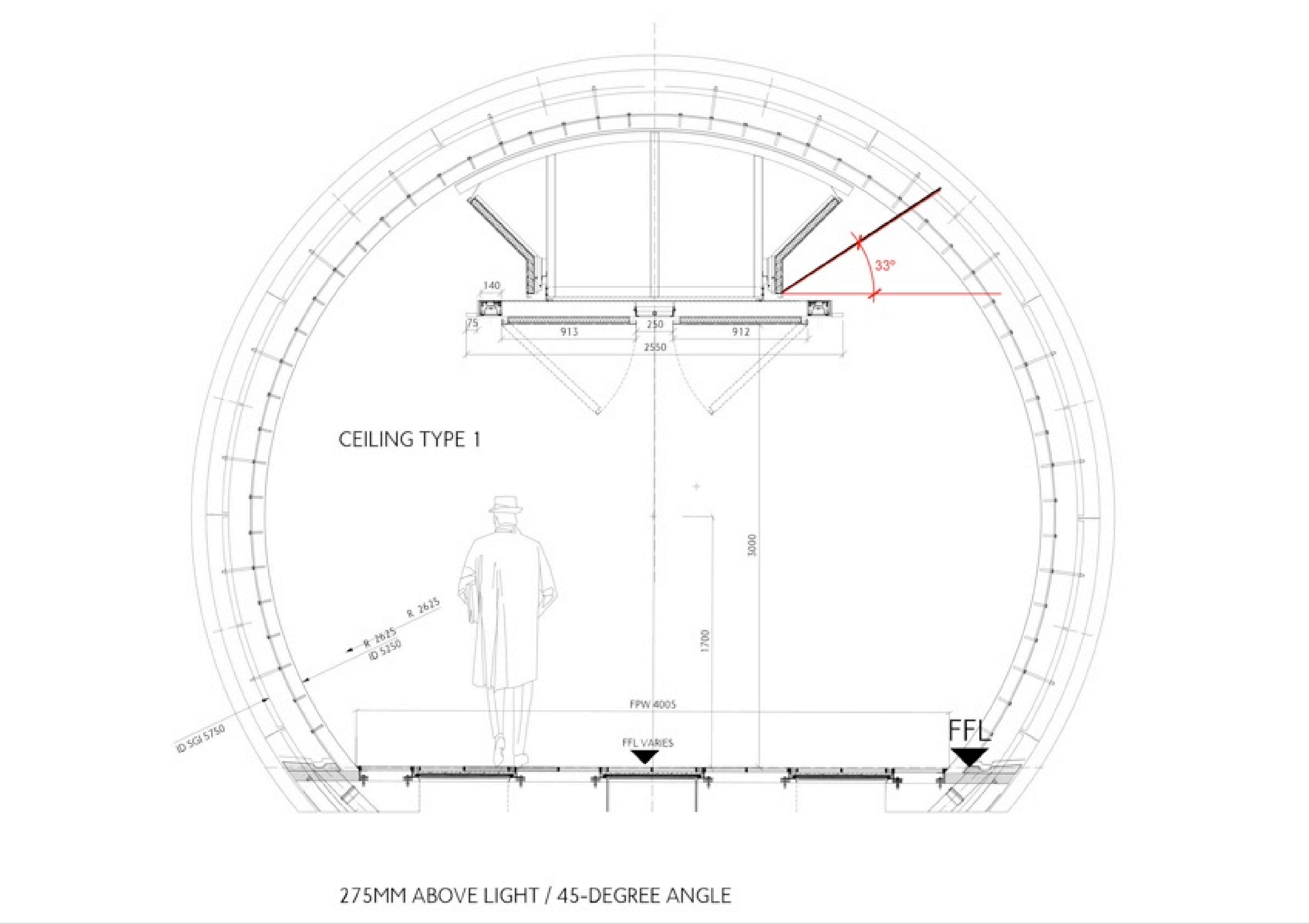
Optimisation of the profile with one variable. Objective function is Illuminance on wall.

King's Cross station cove lighting



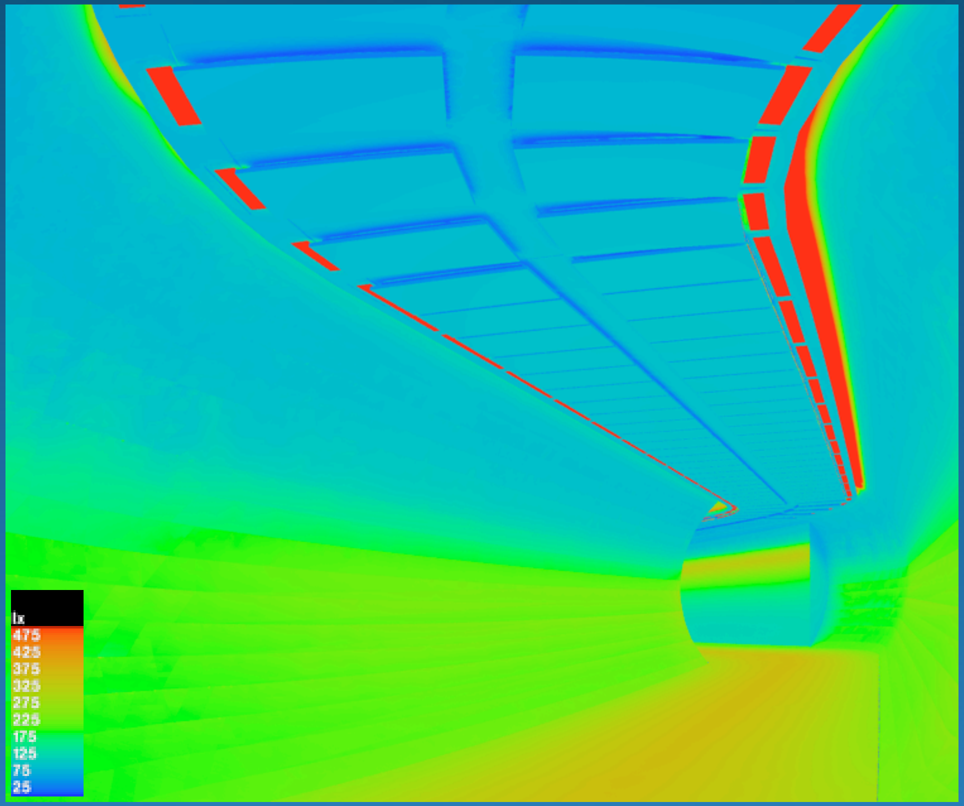
ArupLighting Optimisation of the profile with two variables. Objective function is Illuminance on wall.

King’s Cross station cove lighting

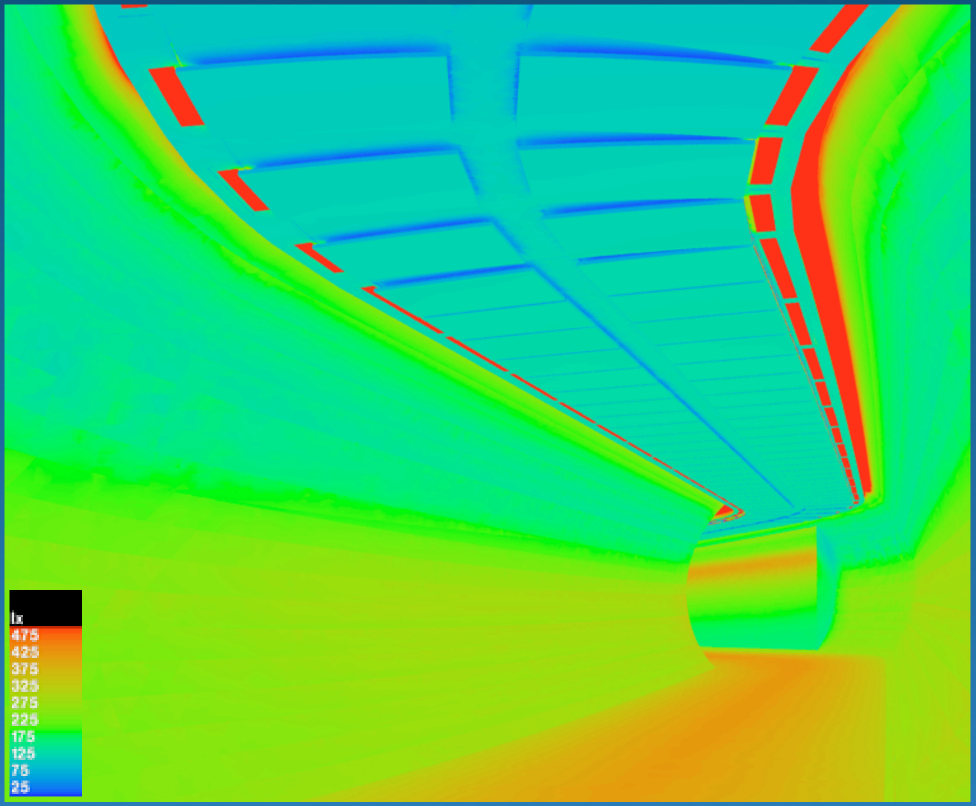


Final optimised solution.

King's Cross station cove lighting



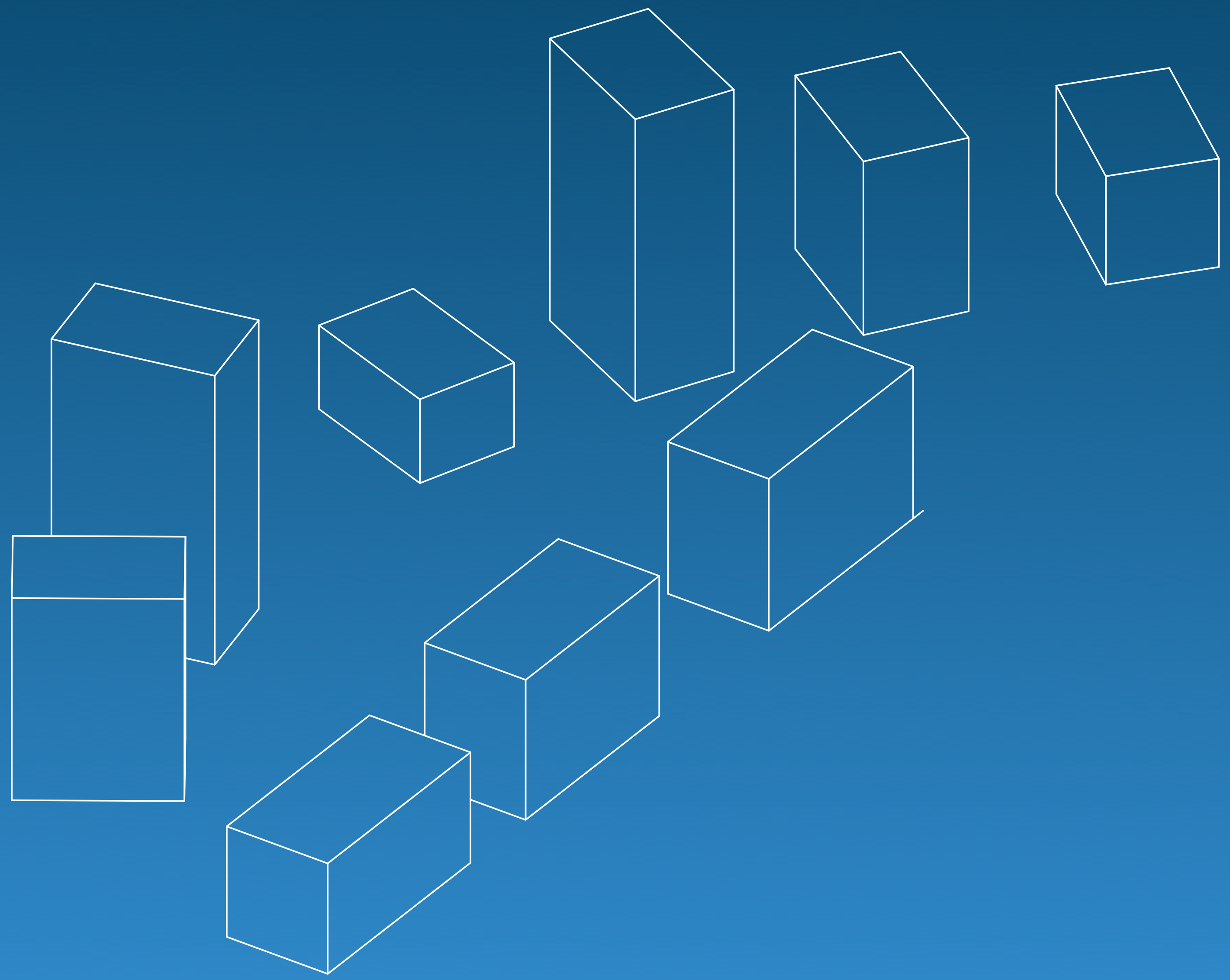
Initial architect proposal



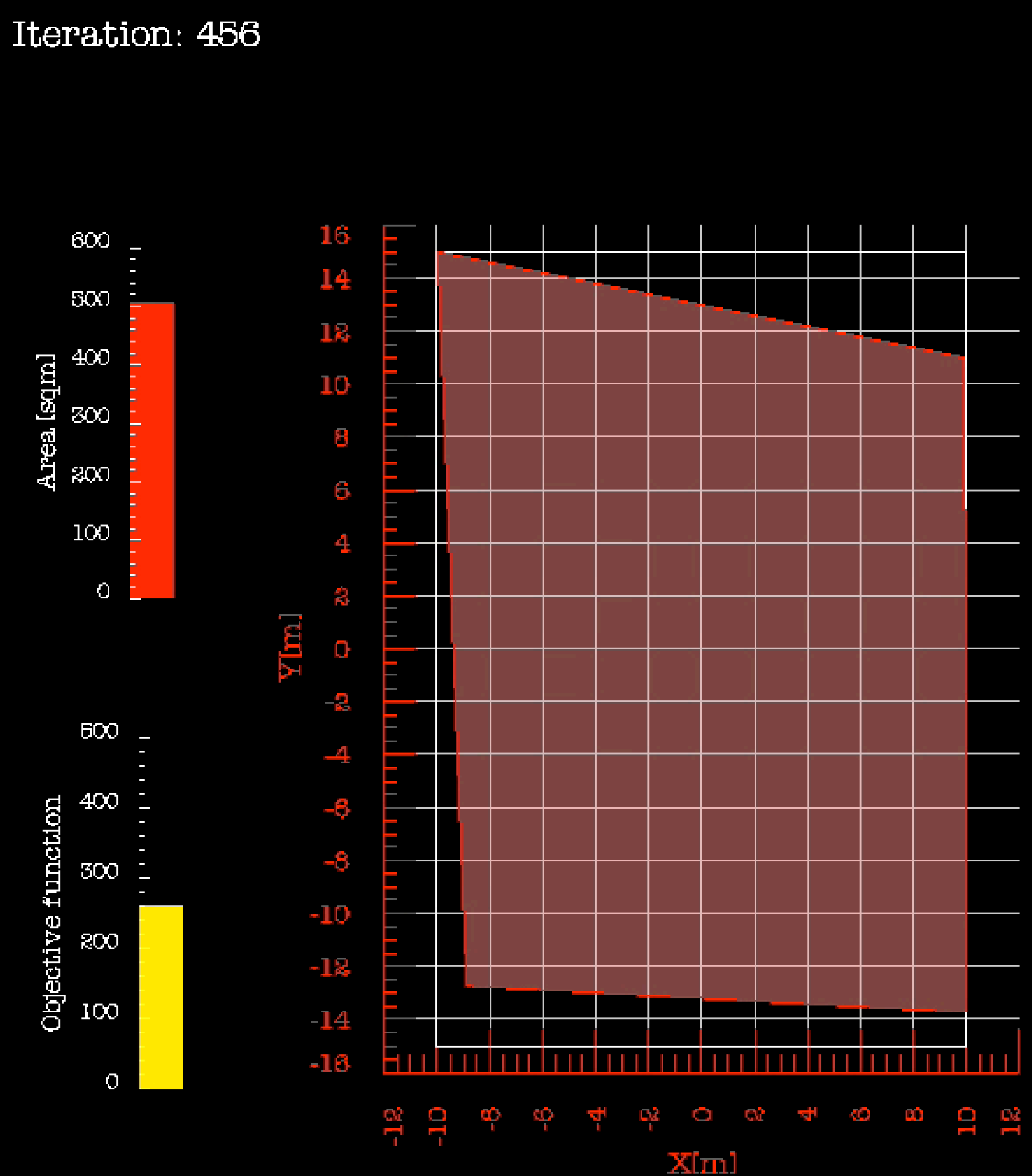
Optimised configuration

Initial proposal vs. final solution performances: chromatic scale images for illuminance.

Tartu residential development

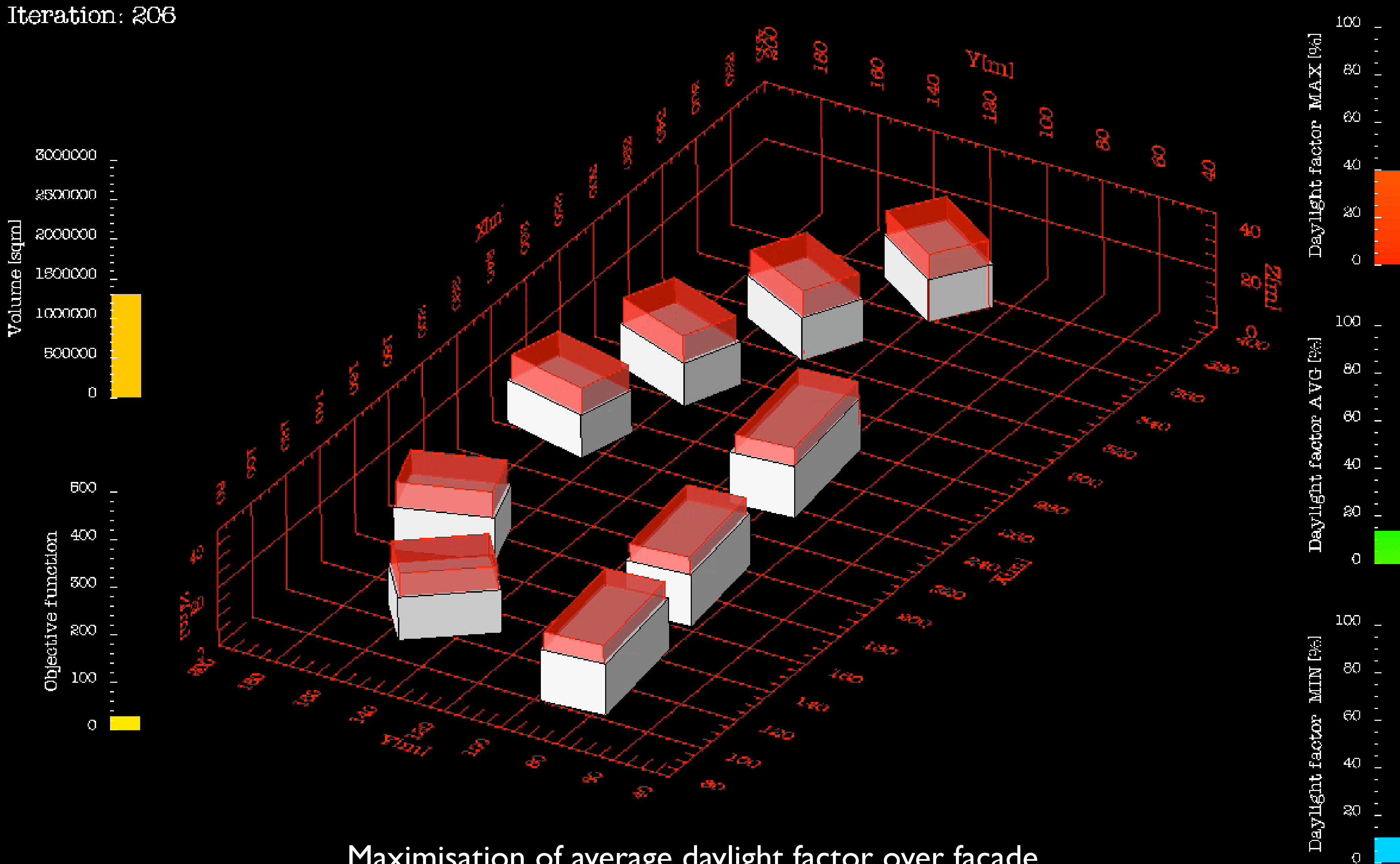


Tartu residential development



Maximisation of site view and floor area
Competition entry with Gianni Botsford Architects

Tartu residential development

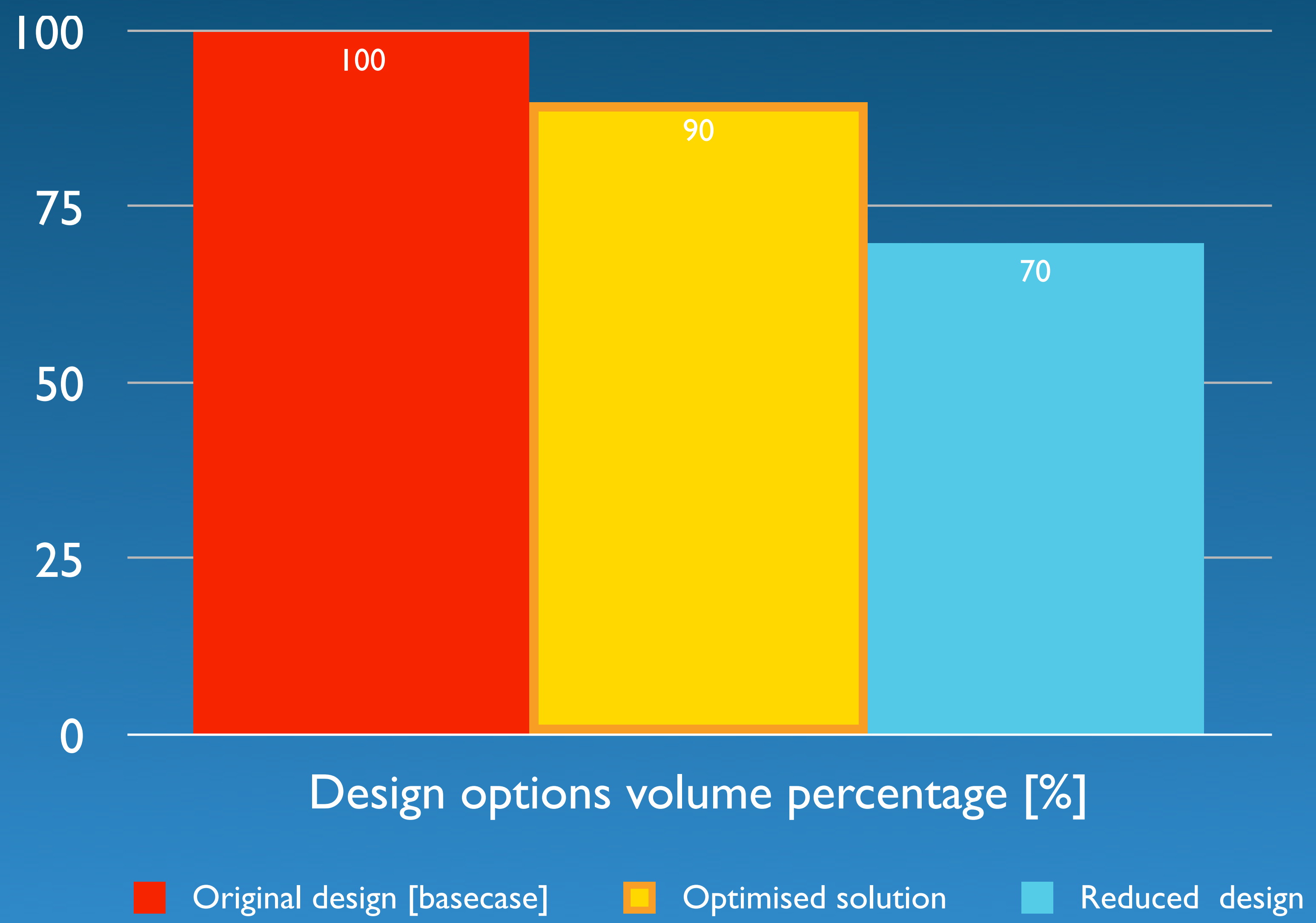


Maximisation of average daylight factor over facade
and volume of development.
Competition entry with Gianni Botsford Architects

Lewisham Gateway development

Maximisation of daylight availability for existing building
Minimisation in the reduction of volume to meet the requirement.
This animation is an early stage of the study.

Lewisham Gateway development



Conclusions:

- ☞ Radiance can be used as a calculation engine within an optimisation process
- ☞ Simplex search is quick and easy to set up for problem with smooth objective functions up to 10 variables... but mind the local maxima!
- ☞ Matlab and his pattern search seems to be the robust choice for more complex problems
- ☞ Often the machine needs help: better to find the way to reduce the complexity of a problem before starting to script

Thanks!