
Optimization methods in CFD

An open-source approach using
DAKOTA and OpenFOAM

Joel Guerrero

University of Genova + Wolf Dynamics

- This training material is compatible with OpenFOAM 8 and DAKOTA 6.13.
- You can find the tutorials used during this presentation at the following link, <http://www.wolfdynamics.com/tutorials.html?id=174>
- You can download DAKOTA at the following link: <https://dakota.sandia.gov/>
- If you are an OpenSUSE user you can find general instructions on how to compile DAKOTA at the following link: <https://www.youtube.com/playlist?list=PLoI86R1JVvv-wuvCrpq28pOzS77s1GBz1>

Roadmap to the presentation

- 1. Brief introduction to numerical optimization and optimization in CFD**
- 2. CFD optimization loop – The big picture**
- 3. DAKOTA overview**
- 4. CFD optimization loops in action – Some examples**
- 5. Live demonstration**

1. Brief introduction to numerical optimization and optimization in CFD

What is optimization?

- In plain English, optimization is the act of obtaining the best result under given circumstances.
- The ultimate goal is either to minimize, maximize, or equalized a quantity of interest (QOI).
- In order to optimize a QOI, we should be able to measure it, quantitatively or qualitatively.
- Optimization, in its broadest sense, can be used to solve any real-life or engineering problem, such as,
 - Finance, health, construction, operations, manufacturing, transportation, engineering design, sales, public services, mail, communication networks, energy distribution, delivery services, CFD and so on.

What is optimization?

- If you can measure the QOI, you can optimize it.
- And it does not matter how you measure the QOI.
- To find the optimal solution there are many methods available, in this training we re going to explore a few of them.
- There are also many tools to conduct optimization, we are going to use one them (DAKOTA).

What is optimization?

- Mathematically speaking, an optimization problem can be formulated as follows,

Find $\mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix}$ which **minimizes**
maximizes
equalized $f_j(\mathbf{X}), \quad j = 1, 2, \dots, q$

Design vector

Quantity of interest

Subject to inequalities and equalities constraints (linear and/or non-linear)

$$g_j(\mathbf{X}) \leq 0, \quad j = 1, 2, \dots, m$$

$$l_j(\mathbf{X}) = 0, \quad j = 1, 2, \dots, p$$

where \mathbf{X} is an n -dimensional vector called the design vector, $f_j(\mathbf{X})$ is the cost function, objective function or QOI, and $g_j(\mathbf{X})$ and $l_j(\mathbf{X})$ are known as inequality and equality constraints, respectively.

Basic concepts in optimization

- The biggest hurdle to overcome in numerical optimization is the design vector.

This guy is responsible for the
curse of dimensionality

→

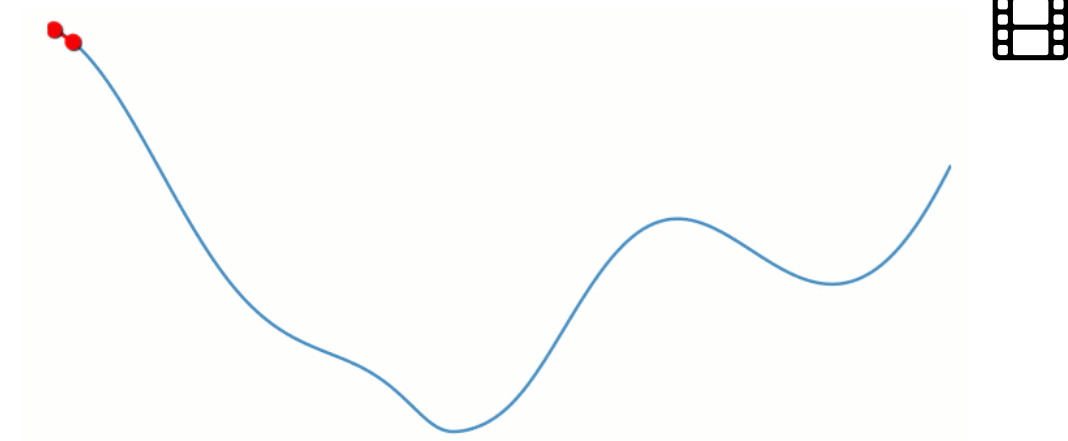
$$\mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix}$$

The curse of dimensionality...

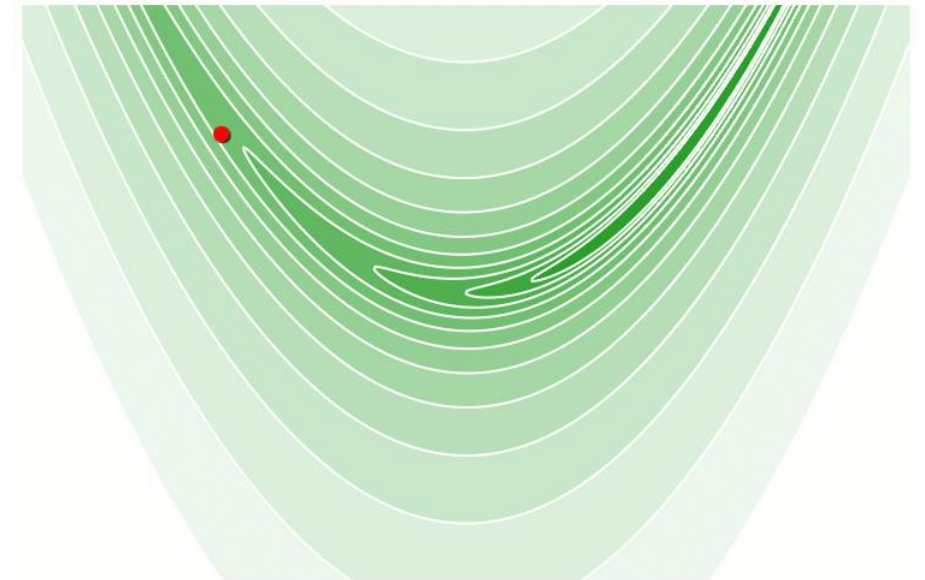
“ The higher the number of design variables in a modelling problem, the more objective function measuring locations we need if we are to build a reasonably accurate predictor [1]. ”

Basic concepts in optimization

- During this training session we will introduce a few basic concepts using univariate and bivariate functions.
- The choice of using univariate and bivariate functions is to help visualize the various concepts.
- However, have in mind that in optimization the design space can be multivariate or n dimensional.
- In the slides to follow, the goal is to optimize a toy function or quantity of interest (QOI).
- This toy function or QOI can easily represent the output or trend of an actual application.



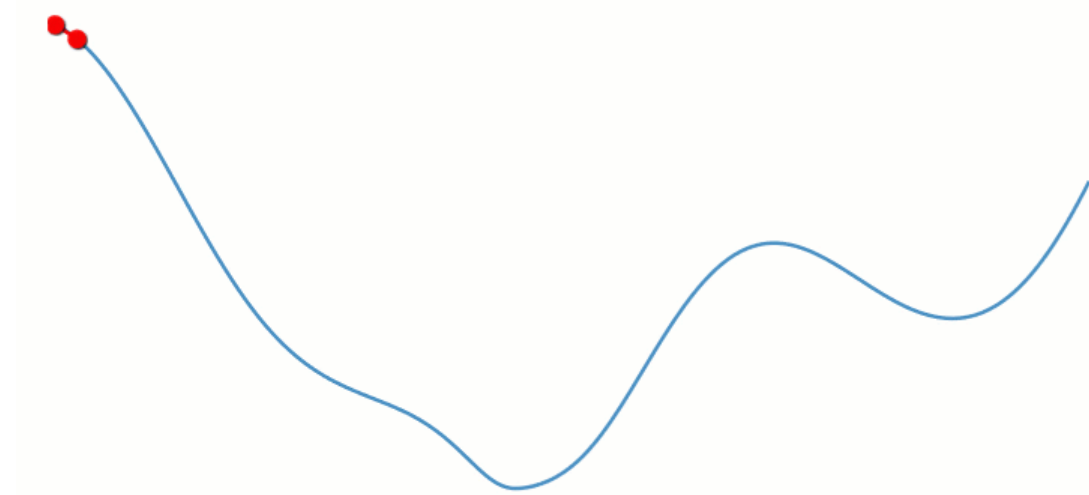
<http://www.wolfdynamics.com/training/opt/ani1.gif>



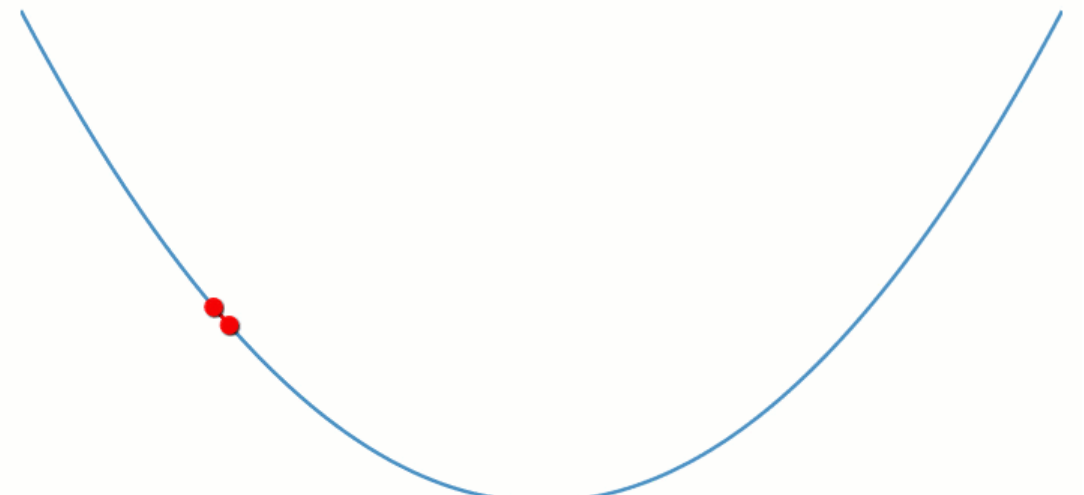
<http://www.wolfdynamics.com/training/opt/image6.gif>

Basic concepts in optimization

- A unimodal function contains one global minimum or maximum.
- A multimodal function contains many global and local minima/maxima.



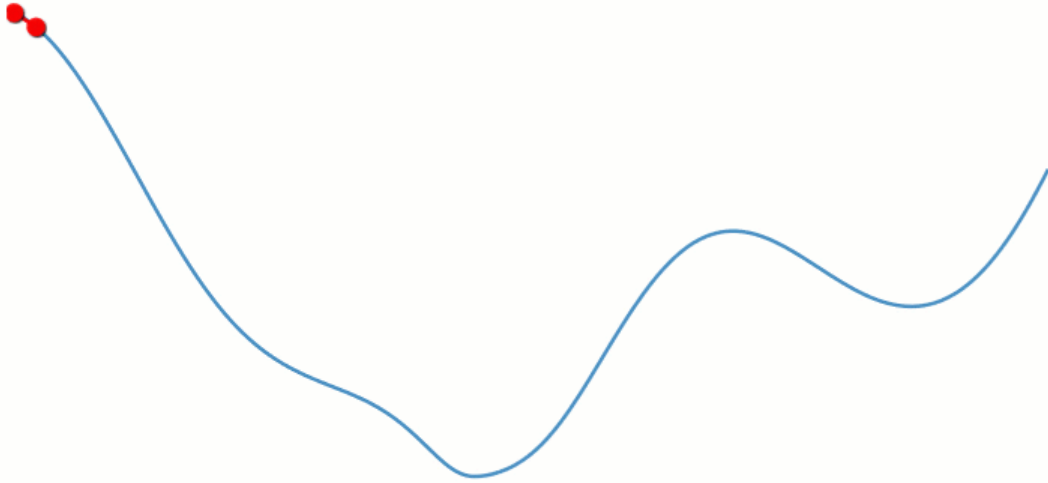
Multimodal function – Many global and local minima/maxima
<http://www.wolfdynamics.com/training/opt/ani1.gif>



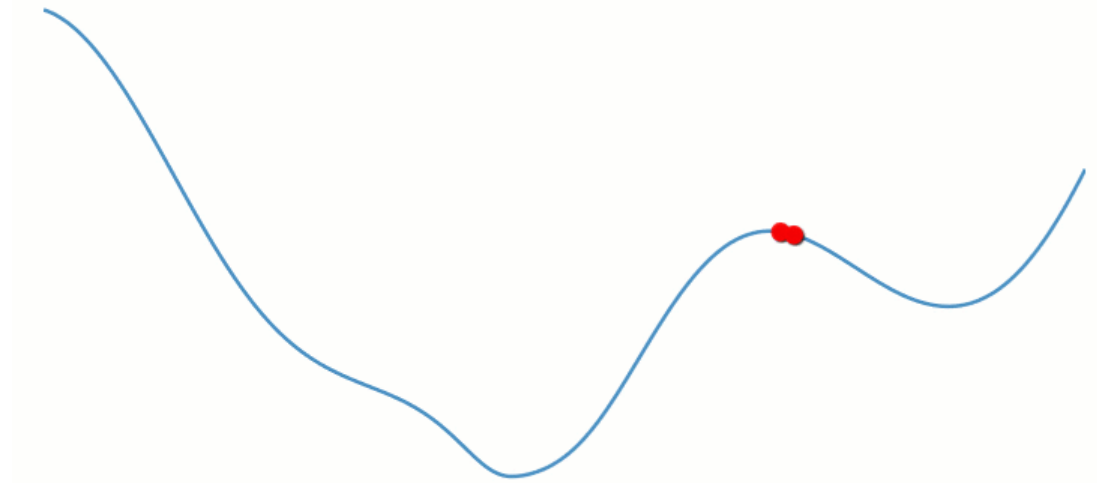
Unimodal function – One global minimum
<http://www.wolfdynamics.com/training/opt/ani2.gif>

Basic concepts in optimization

- Depending on our goal (local or global optimization) or the behavior of the QOI, we should use a particular optimization method.
- Also, depending on the starting point we might arrive to a global or local minima/maxima.



Multimodal function – Global minimum
<http://www.wolfdynamics.com/training/opt/ani1.gif>

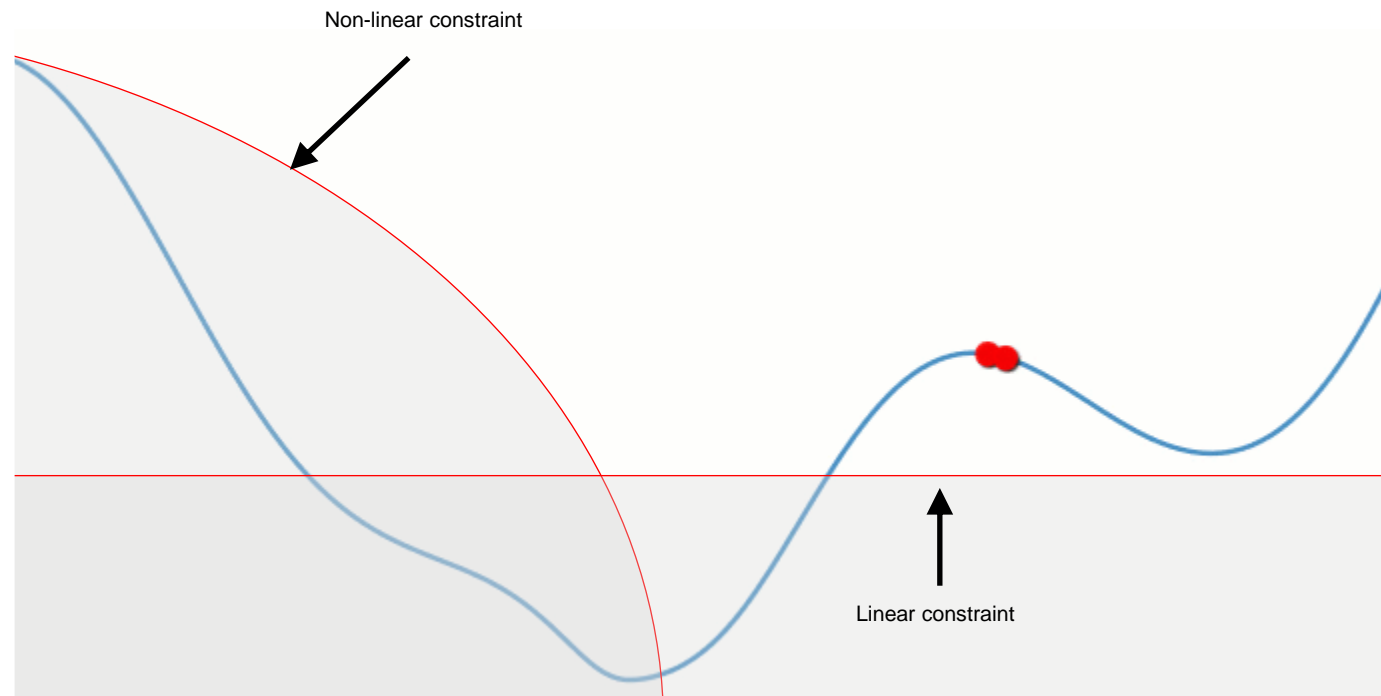


Multimodal function – Local minimum
<http://www.wolfdynamics.com/training/opt/ani3.gif>



Basic concepts in optimization

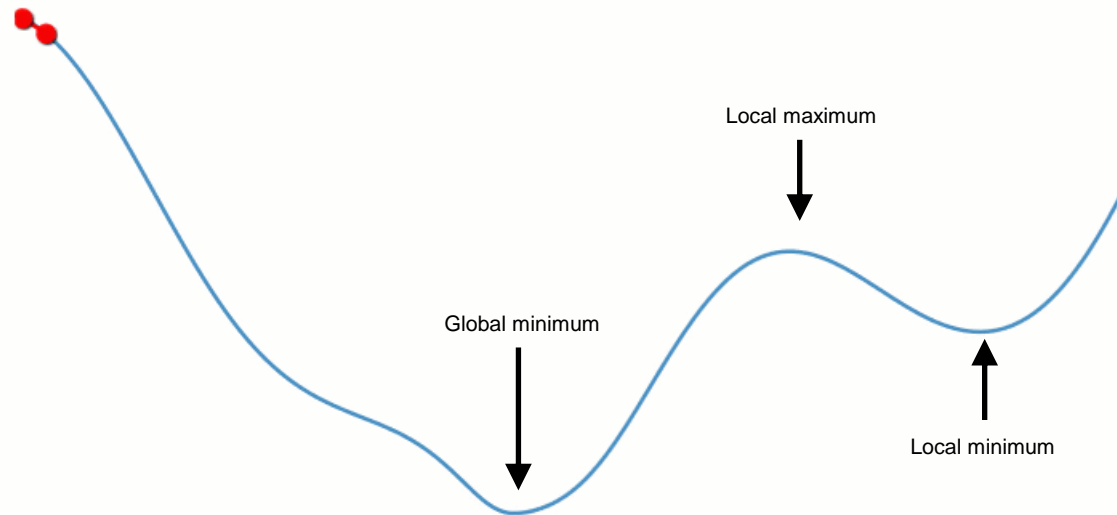
- The design space can be bounded or unbounded.
- We can optimize this function or QoI, subject to many linear and/or non-linear constraints (equalities and inequalities).



Constrained multimodal function – The shaded area represents the non-feasible region

Basic concepts in optimization

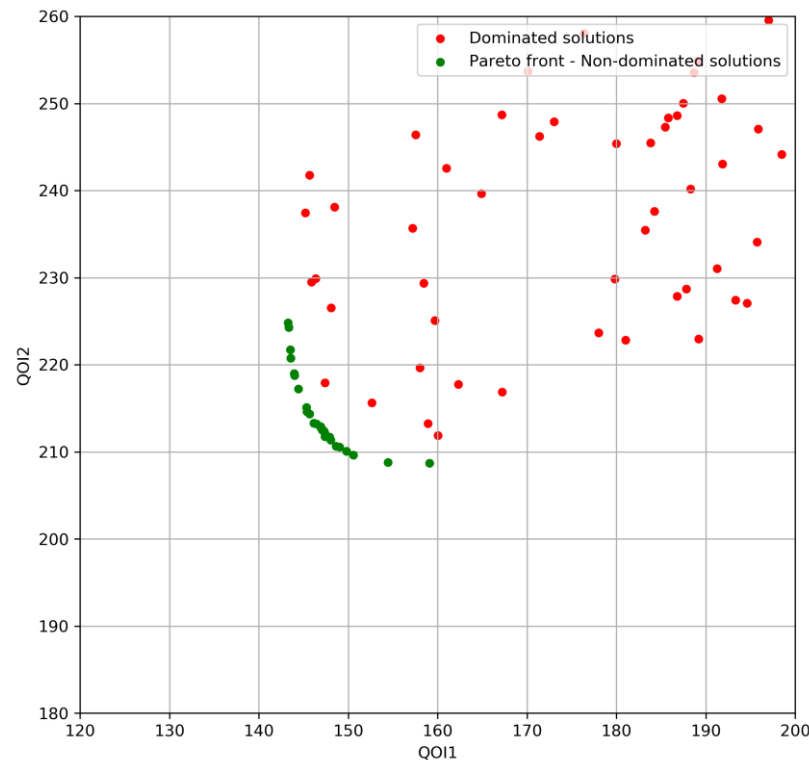
- We can also deal with single-objective and multi-objective optimization.
- In single-objective optimization we are interested in optimizing one objective function.
- The optimization problem can be bounded and constrained.
- The optimization method can converge to a local or global optimal value.
- The optimal value depends on the starting point of the search algorithm.



Multimodal function – Many global and local minima/maxima
<http://www.wolfdynamics.com/training/opt/ani1.gif>

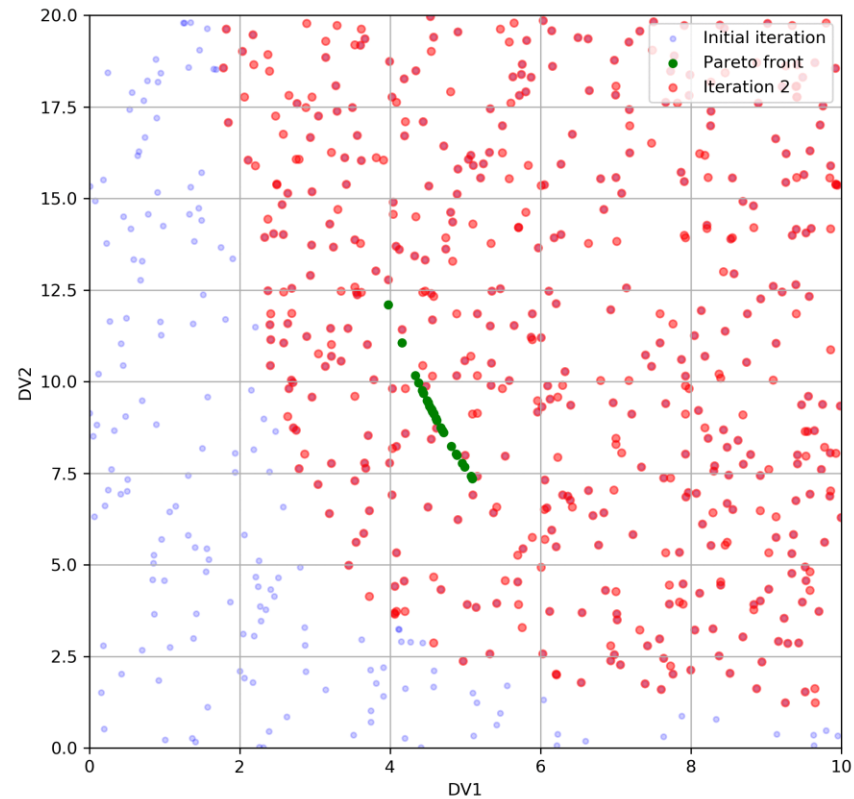
Basic concepts in optimization

- We can also deal with single-objective and multi-objective optimization.
- In multi-objective optimization we are interested in optimizing more than one objective function or QOI simultaneously.
- The final goal is to find a representative set of optimal solutions (Pareto frontier or non-dominated solutions), quantify the trade-offs in satisfying the different objectives, and/or finding a single solution that satisfies the subjective preferences of a human decision maker.



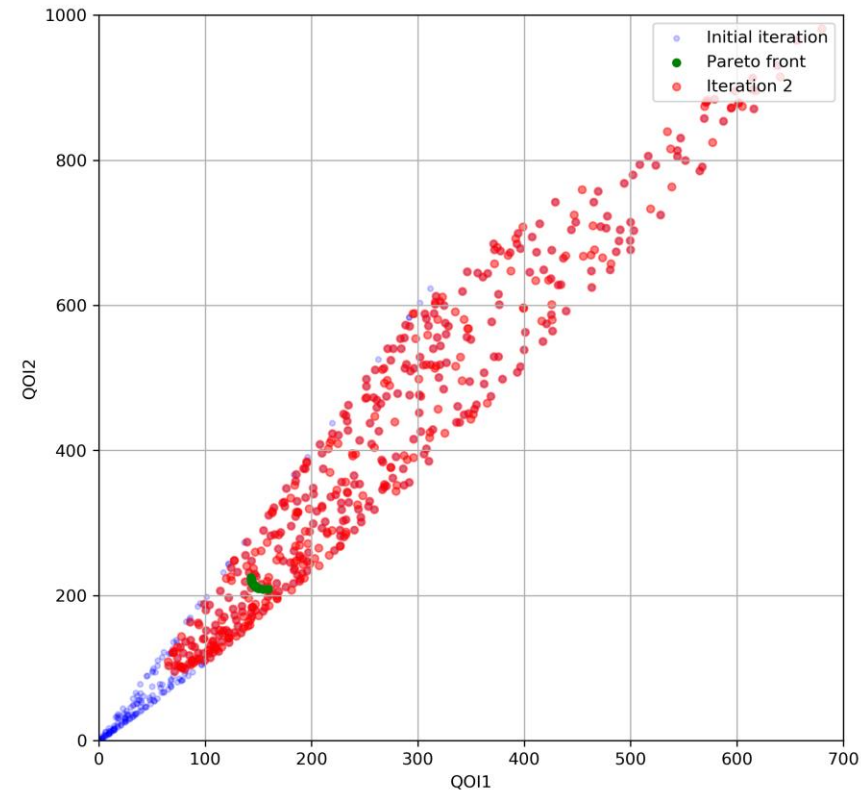
Basic concepts in optimization

- Each optimal solution in the objective function space can be mapped to the design space.
- The functions to optimize can be constrained (linear and non-linear constraints).
- The design space can be bounded or unbounded.



Design space

<http://www.wolfdynamics.com/training/opt/ani4.gif>



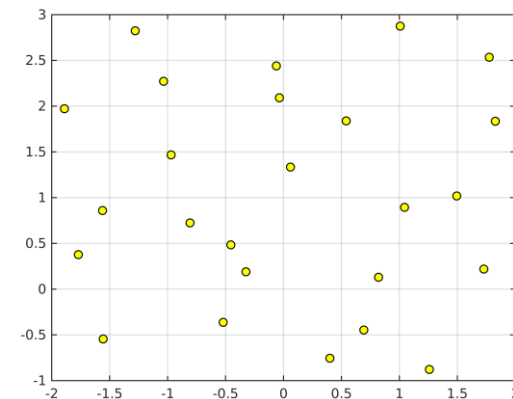
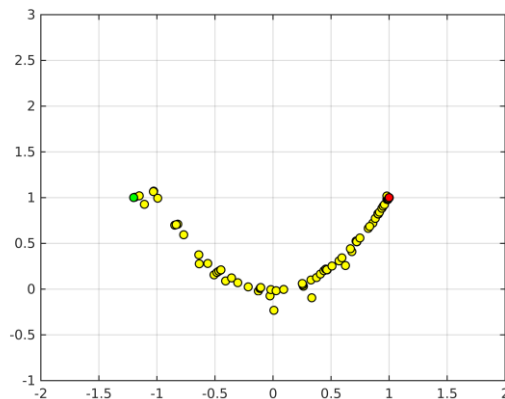
Objective function space

<http://www.wolfdynamics.com/training/opt/ani5.gif>



Brief overview of optimization methods

Design optimization (DO)	Design space exploration (DSE)
<ul style="list-style-type: none">• Converging-Iterative process.	<ul style="list-style-type: none">• Diverging-Iterative process.
<ul style="list-style-type: none">• DO aims at determining the optimum design.	<ul style="list-style-type: none">• DSE aims at searching and characterizing the design space.
<ul style="list-style-type: none">• DO strategies have two distinct parts; formulate the problem and converge to the solution.	<ul style="list-style-type: none">• Once we know the design space, a better solution can then be found, for example, by using DO.
<ul style="list-style-type: none">• DO depends on a well-posed optimization problem formulation (starting point, gradients computation, tolerances, and so on).	<ul style="list-style-type: none">• Contrary to DO, in DSE we do not need a well formulated problem. It is enough to define an efficient sampling plan.
<ul style="list-style-type: none">• DO relies in gradient-based methods and derivative-free methods.	<ul style="list-style-type: none">• DSE relies in sampling the design space using design of experiments methods* (e.g., space filling, full factorial).

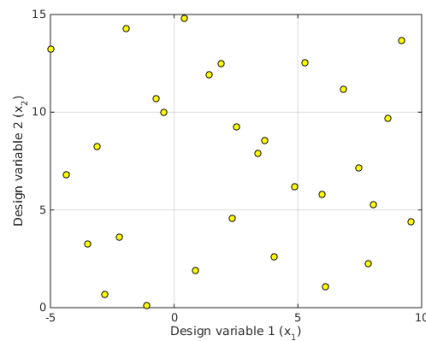


* And the more modern design and analysis of computer experiments (DACE) methods.

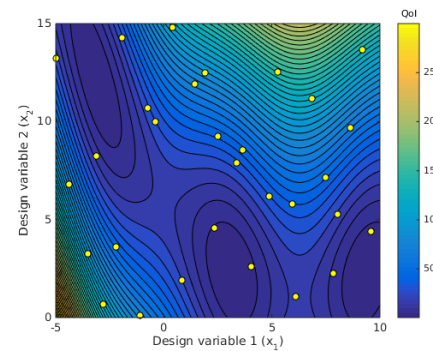
Brief overview of optimization methods

Surrogate-based optimization (SBO)

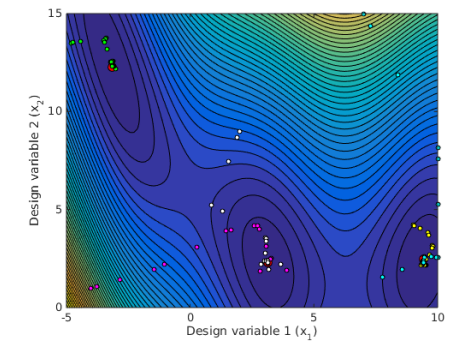
- It is a mix of DSE and DO (space exploration and converging-iterative process).
- SBO explores the design space from a limited number of observations (it can be used with high multi-dimensional design spaces).
- Then, SBO exploits and optimizes the design space by constructing a surrogate model (also known as meta-model, predictor model, or response surface).
- At the surrogate level, any optimization method can be used (gradient-based or derivative-free). Working at the surrogate level is orders of magnitude faster than working at the high-fidelity level.
- SBO is well fitted to engineering design. Especially during the conceptual and preliminary design phases or to construct digital twins.



Explore the design space



Construct the surrogate



Exploit and optimize at the surrogate level

Brief overview of optimization methods

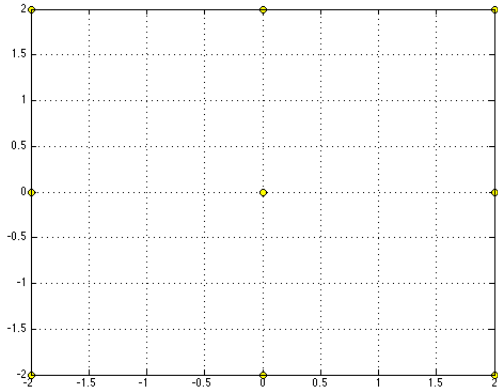
- The main idea of **design space exploration** is to search the design space in a very efficient way at a minimal cost.
- In **design space exploration**, we follow a systematic mathematical or statistical approach to acquire model behavior to the maximum extent.
- With design space exploration we can:
 - Gain a deep statistical understanding of the problem.
 - Explore a wide design space through intelligent sampling.
 - Identify the most important influencing design variables.
 - Create accurate mathematical models.
 - Provide a set of starting points for design optimization.

Brief overview of optimization methods

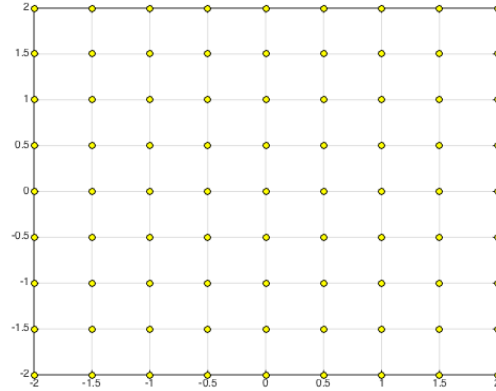
- **Design space exploration** can be conducted with parametrical or multi-dimensional studies, and/or with computer experiments (e.g., DACE).
- Parametrical studies explore the effect of parametric changes within simulation models by computing response data sets at a selection of points in the parameter space, yielding one type of sensitivity analysis. The selection of points is deterministic and structured, or user-specified.
- Classical design of experiments (DoE) methods and the more modern design and analysis of computer experiments (DACE) methods are both techniques which seek to extract as much trend data from a parameter space as possible using a limited number of sample points.
- In DACE, the sampling is stochastic and covers most of the design space (space filling experiments).
- A few DACE sampling techniques:
 - Orthogonal array sampling, Latin hypercube sampling, Quasi-Monte Carlo sampling.

Brief overview of optimization methods

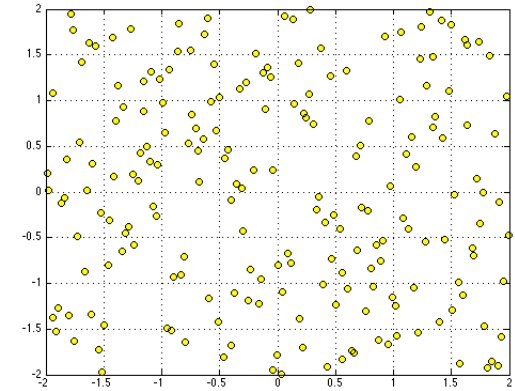
- Different experiments designed using **design space exploration** methods.



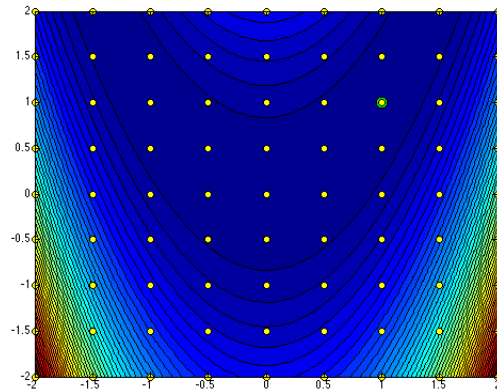
DOE experiment



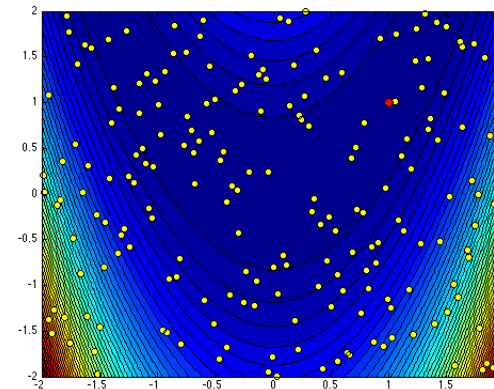
Multidimensional study



DACE experiment



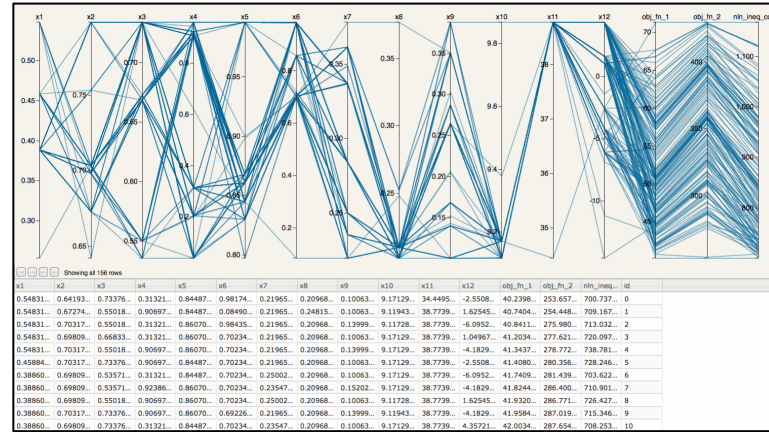
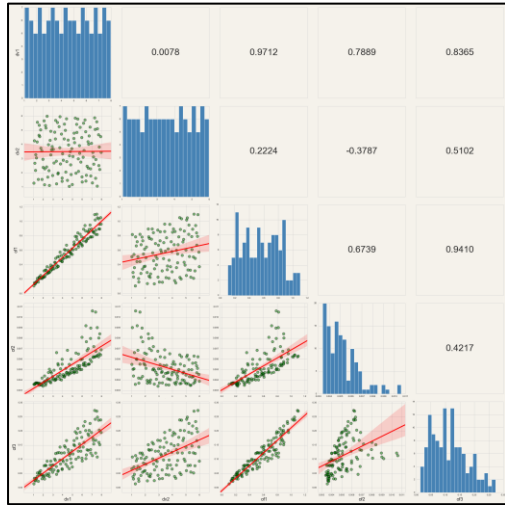
Response surface constructed using multidimensional study



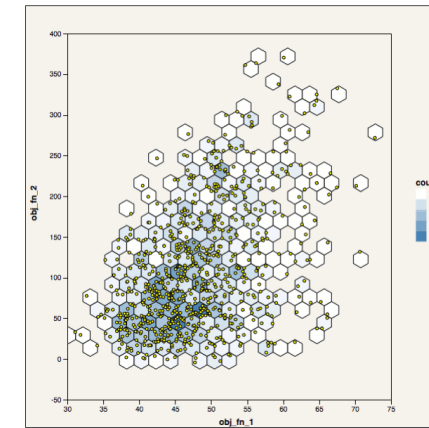
Response surface constructed using DACE sampling

Brief overview of optimization methods

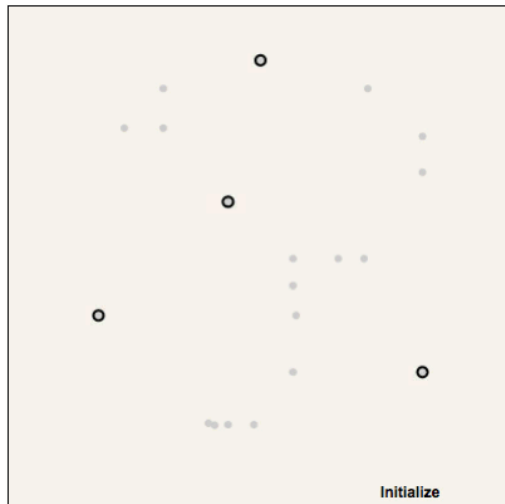
- We can use the information obtained from the **design space exploration** study to extract knowledge using data visualization, machine learning and statistical analysis.



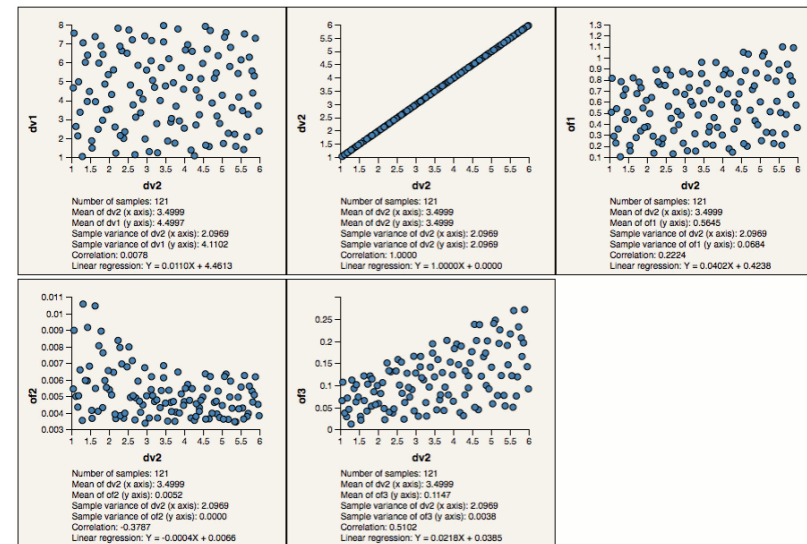
<http://www.wolfdynamics.com/training/opt/image3.gif>



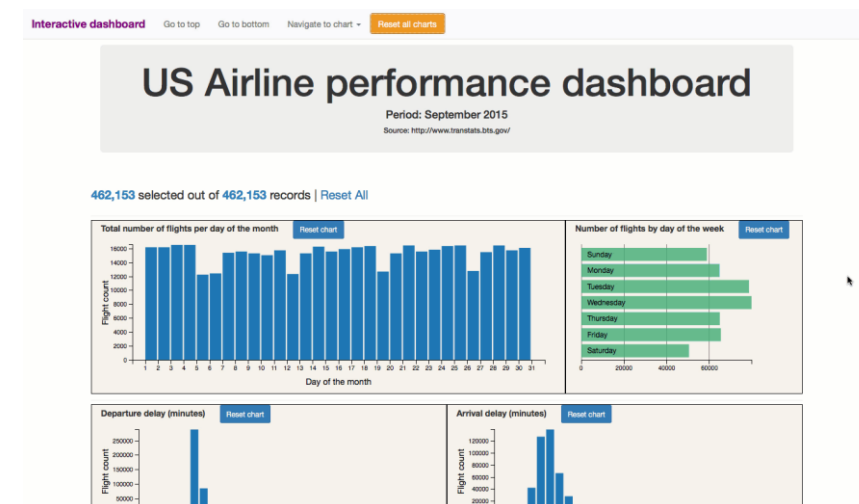
<http://www.wolfdynamics.com/training/opt/image1.gif>



<http://www.wolfdynamics.com/training/opt/image9.gif>



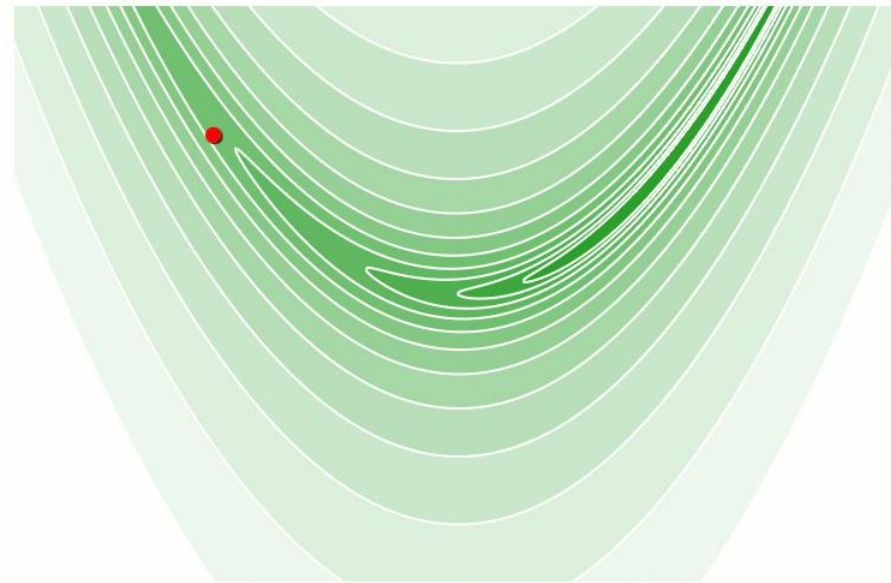
<http://www.wolfdynamics.com/training/opt/image2.gif>



<http://www.wolfdynamics.com/training/opt/image5.gif>

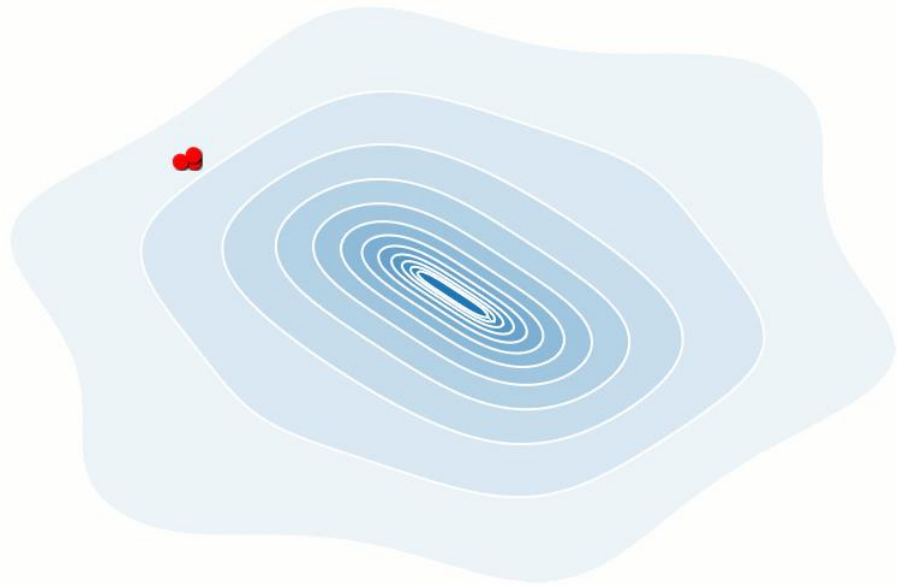
Brief overview of optimization methods

- **Gradient-based optimization methods** looks for improvement based on derivative information.
- These methods are highly efficient, specially if you are dealing with local optimization. They have the best convergence rates of all of the optimization methods.
- They do not perform well with global optimization or noisy outcomes.
- They are the clear choice when the problem is smooth, unimodal, and well-behaved.
- However, when the problem exhibits a non-smooth, discontinuous, or multimodal behavior, these methods can also be the least robust since inaccurate gradients will lead to bad search directions.

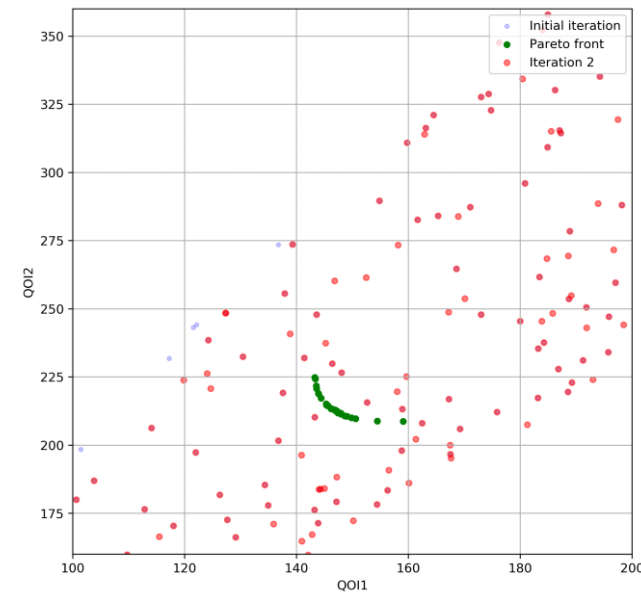


Brief overview of optimization methods

- **Derivative-free methods** (global and local), do a broad exploration of the design space with selective exploitation.
- They can be used for local and global optimization.
- They exhibit much slower convergence rates for finding an optimum, and as a result, tend to be much more computationally demanding than gradient-based methods.
- But they deserve consideration when the problem is non-smooth, multimodal, poorly behaved, or the derivative are expensive to compute.
- If you are dealing with multi-objective optimization, you should use derivative-free methods.



<http://www.wolfdynamics.com/training/opt/image7.gif>



<http://www.wolfdynamics.com/training/opt/ani5.gif>

Brief overview of optimization methods

- Choosing and optimization method – General guidelines – Decision matrix

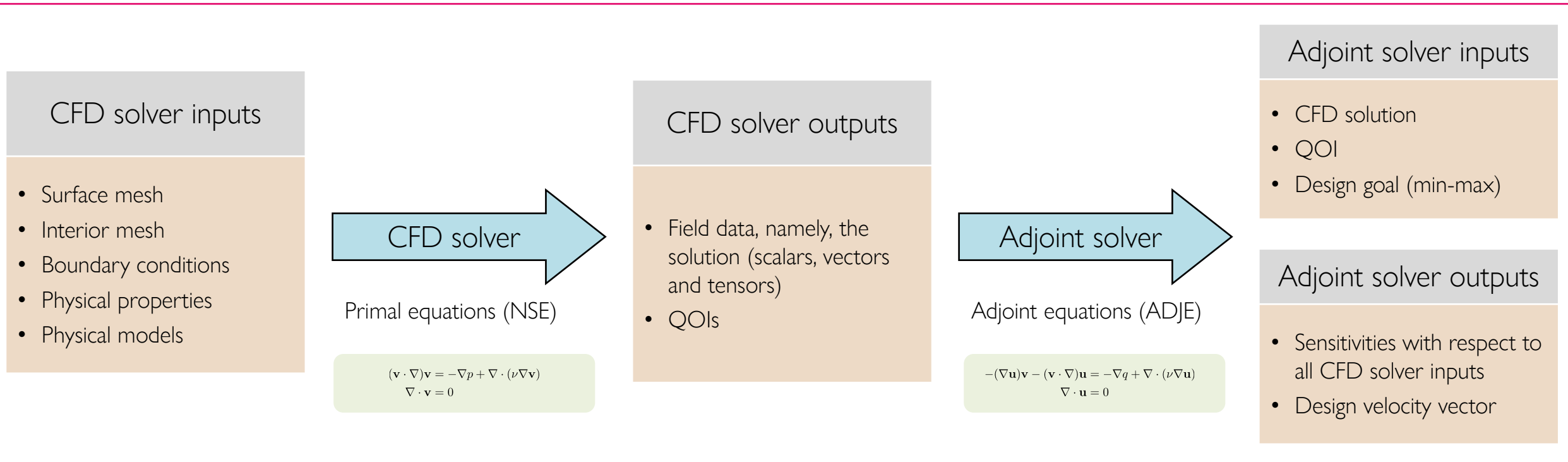
	Unconstrained or bound-constrained problems	Nonlinearly-constrained problems
Smooth and inexpensive	Any method. Gradient-based will be the fastest	Gradient-based methods
Smooth and expensive	Gradient-based methods	Gradient-based methods
Non-smooth and inexpensive	Derivative-free methods Surrogate based optimization	Derivative-free methods Surrogate based optimization
Non-smooth and expensive	Surrogate based optimization	Surrogate based optimization
Multi-objective	Derivative-free methods Surrogate based optimization	Derivative-free methods Surrogate based optimization

Brief overview of optimization methods

- **Gradient-based methods** and **derivative-free methods** find the optimal value by computing the sensitivity of the QOI in response to changes in each design variable.
- For problems when we have a large design vector, this might not be the best way to compute the sensitivities.
- Recall the curse of dimensionality.
- A way to circumvent the curse of dimensionality is by computing the sensitivities by using the adjoint method.
- By sensitivities we mean the response of the QOI to changes of the inputs of the system.
- It is important to stress that this training is not about the adjoint method.
- This training is about the previous techniques, namely, **DO** using gradient-based methods and derivative-free methods, **DSE**, and **SBO**.

Brief overview of optimization methods

- In the framework of CFD, an adjoint solver takes a flow solution and calculate the sensitivity of the QOI in response to all the inputs of the system, simultaneously, in a single computation.
- The QOI could be a measure of the system performance such as the lift or drag of a body, heat flux from a surface or the total pressure drop through a system.



- We can iterate manually or using gradient-based methods.
- Many design iterations might be required to reach a good design or what we are aiming for.

Brief overview of optimization methods

- Adjoint solvers can be:
 - Continuous solvers
 - Discrete solvers
 - Automatic differentiated solvers (algorithmic differentiation)
- Each solver type has different implementation details, limitations, and pros/cons, but at the end of the day, they all find derivatives with respect to the shape of the body or flow path, allowing the sensitivities to be evaluated.
- Obtaining an adjoint solution (sensitivities, derivatives, gradients, you name it) gives designers key information on how to modify the shape of the body.
- An adjoint solution can be used to estimate the effect of a change prior to actually making the change.
- To get an adjoint solution, about the same computational resources as for the flow solution are required.
- Some adjoint solvers can be memory eager.

Brief overview of optimization methods

- The adjoint method is a very powerful technique for shape optimization in CFD.
- However, is not an entry level method. It requires a well prepare user.
- It is recommended to use this method for fine tuning and not during the initial stage of product development.
- Have in mind that it can give you very unrealistic shapes that still are optimal.
- We will not address the adjoint method during this training.
- But if you are interested in using it in OpenFOAM, there are a few open-source implementations available. Just no name a couple:
 - Discrete adjoint:
 - <https://github.com/mdolab/dafoam>
 - Algorithmic differentiation:
 - <https://www.stce.rwth-aachen.de/research/software/discreteadjointopenfoam>
 - Continuous adjoint
 - <https://www.openfoam.com/releases/openfoam-v2006/numerics.php>

Brief overview of optimization methods

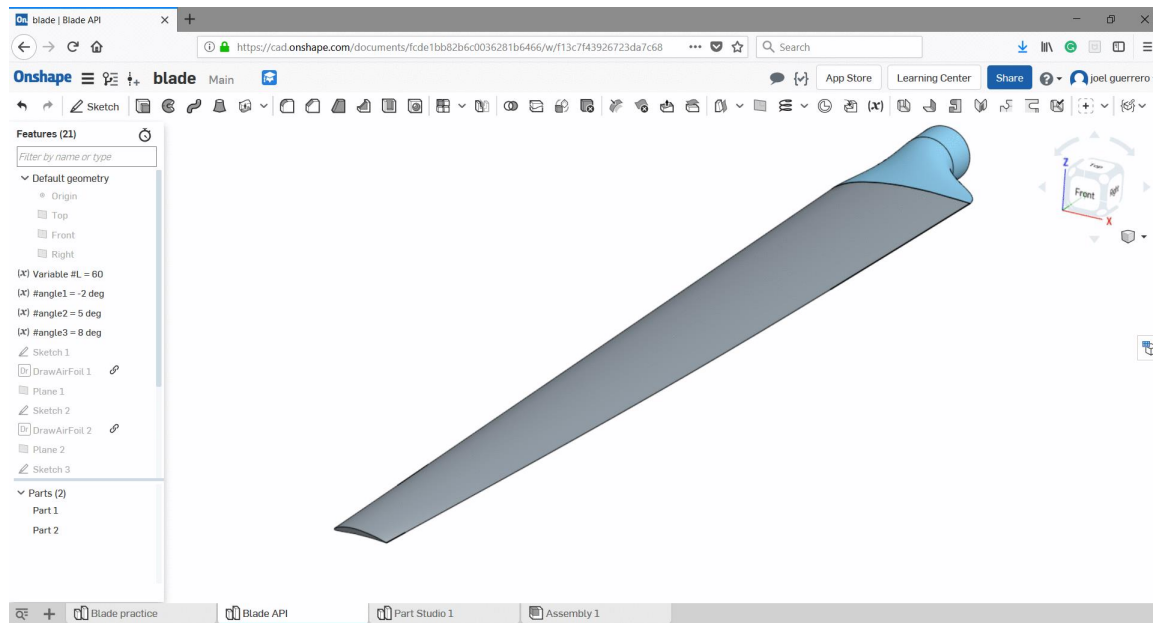
- Adjoint solvers are closely related to mesh morphing and free-form deformation techniques.
- Shape sensitivity data can be combined with mesh morphing to guide smooth mesh deformations.
- Mesh morphing and free-form deformation are powerful tools that allows designers to alter the geometry at the mesh level to evaluate effects of the design alterations.
- Designers can then iterate to achieve an optimum design without the need to return to the original CAD geometry.
- When using mesh deformation and free-form deformation, we lose the CAD parametrization, and we might get very unrealistic designs (that still are optimal) but are not feasible due to operational or manufacturing requirements.
- For complex geometries, the insight gained by adjoint solvers can take the design in unexpected directions.

Parameter-based optimization and parameter-free optimization

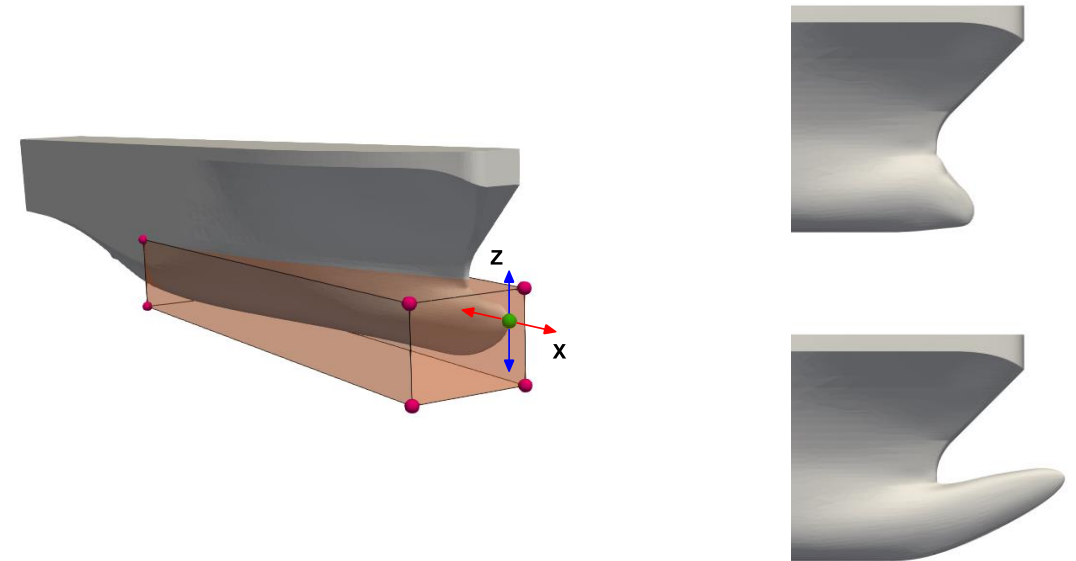
- We can also classify the optimization methods used in CFD according to the geometry parametrization level:
 - Parameter-based or CAD based optimization.
 - Parameter-free or free-form optimization.



<http://www.wolfdynamics.com/training/opt/ani7.gif>



Parameter-based or CAD based optimization



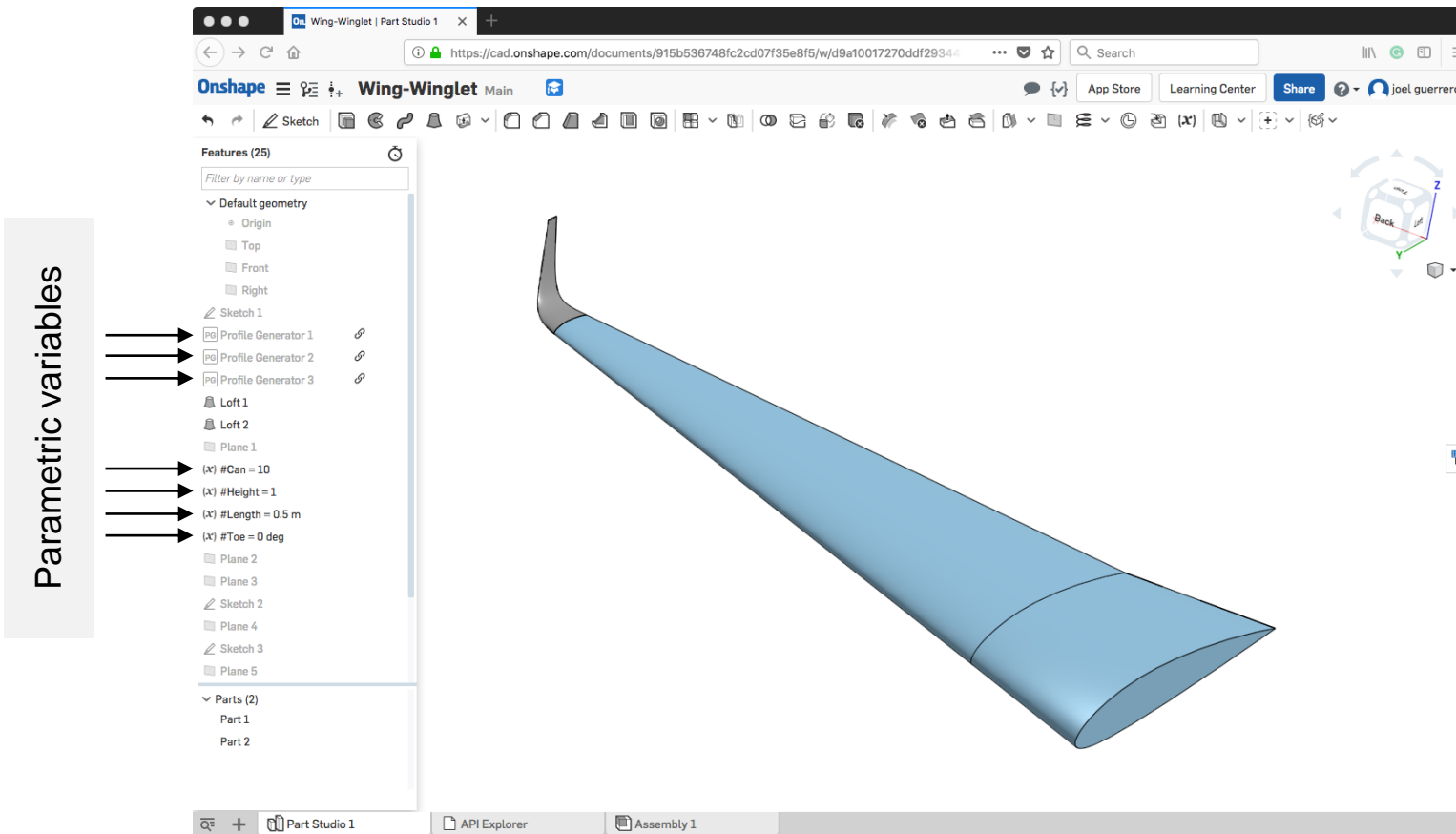
Parameter-free or free-form optimization

Parameter-based optimization – CAD based optimization

- Parameter-based optimization, works at the CAD level.
 - Gives the designer incredible level of control over the geometry.
 - A couple of parametrical variables are enough to make significant and well controlled changes in the final geometry.
 - Changes can be introduced easily.
 - The final geometry is ready to use for manufacturing or production.
 - The main difficulty is making the CAD application interact with the optimization loop.
 - It is usually used with gradient-based and derivative-free methods (local and global).
 - It is a very mature method and widely used in industry.

Parameter-based optimization – CAD based optimization

- Parameter-based optimization is usually used with gradient-based and derivative-free optimization methods.
- It can be used with multi-objective and multi-disciplinary optimization.



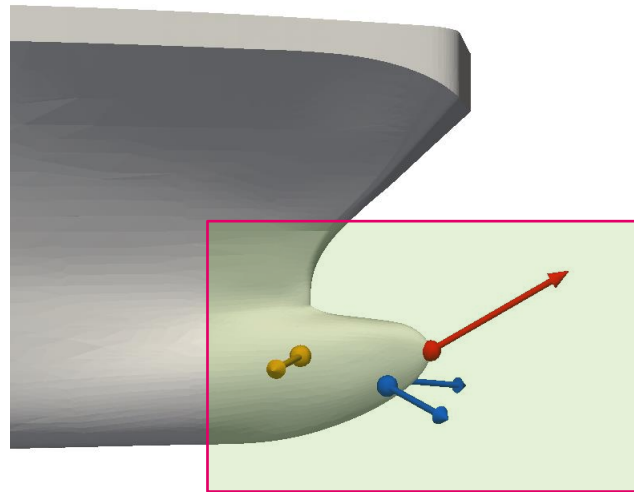
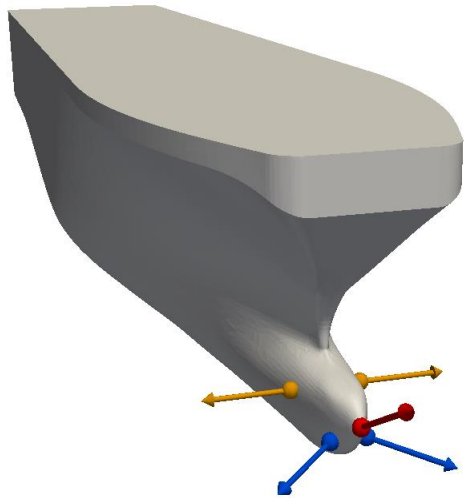
Cant angle and toe angle variations

Parameter-free optimization – Free form deformation

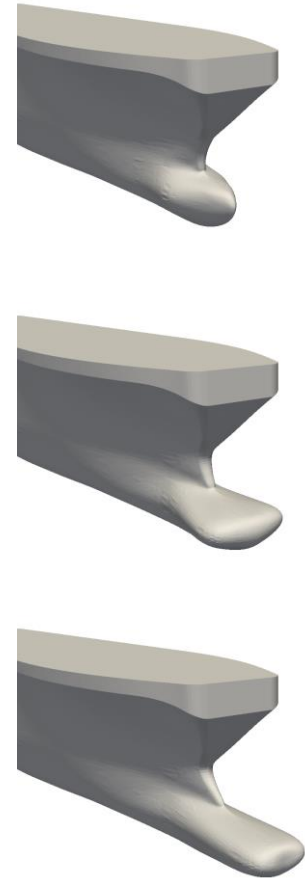
- Parameter-free optimization, works at the surface mesh level or volume mesh level.
 - As is not based on parametrical variables gives a lot flexibility when deforming the geometry.
 - However, the flexibility gained does not necessarily means that the designer has extensive control on the mesh deformation.
 - It requires the selection of many control points or lattice boxes with many control points in order to define deformations.
 - As it can used at the mesh level, it does not require remeshing, reducing in this way the simulation time.
 - But for large mesh deformation, the quality of the mesh can be compromised.
 - It is usually used with adjoint methods.
 - The adjoint method has been proved a very efficient way to compute the sensitivities, but they are not easy to use.
 - A lot of R&D is being done and it has been used with success in very specific industries.

Parameter-free optimization – Free form deformation

- Parameter-free optimization is usually used with adjoint optimization methods.
- Difficult to use with multi-objective and multi-disciplinary optimization.
- It is not easy to control, and it can generate unrealistic geometries.



Control points and control box selection



Parameter-based and parameter-free optimization – Summary

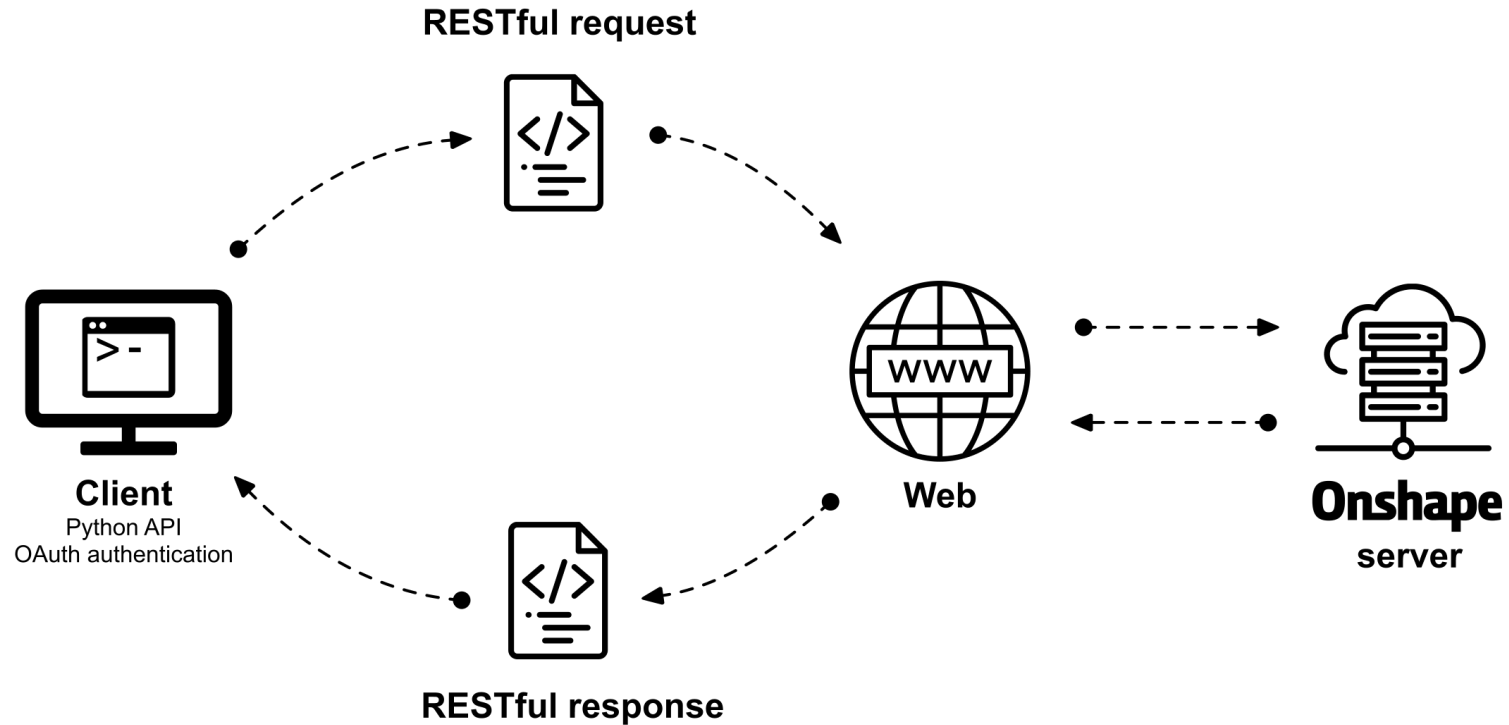
Optimization approach		Pros	Cons	Design stage
<ul style="list-style-type: none"> Parameter-free 	<ul style="list-style-type: none"> Topology optimization. Surface optimization. Both methods are based on the adjoint. 	<ul style="list-style-type: none"> Can generate innovative and unconventional designs. Fast, it requires one solver run or a few design iterations. Gives a lot insight on where to modify the geometry. 	<ul style="list-style-type: none"> Topology optimization (volume based): <ul style="list-style-type: none"> Requires reverse engineering the results into CAD. Used only for internal flows. Adjoint optimization (surface based): <ul style="list-style-type: none"> Confined to small changes, unless used in an iterative way. Can be difficult to interpret the results. Limited to the implemented QOIs. Difficult to use in multi-objective and multi-disciplinary optimization. 	<ul style="list-style-type: none"> Topology optimization can be used from Initial design to fine tuning. Adjoint optimization is preferably used for fine tuning the design.
<ul style="list-style-type: none"> Parametric-based 	<ul style="list-style-type: none"> Fully parametric CAD. 	<ul style="list-style-type: none"> CAD geometry of high quality. Can be used with gradient-based and derivative-free methods (local and global). Applicable to multi-objective and multi-disciplinary optimization. Gives insight, the designer can determine which parametric variables have higher correlations. 	<ul style="list-style-type: none"> It requires many solver runs in order to compute the sensitivities. Confined to the design space of parametric model. Requires a parametric CAD model. 	<ul style="list-style-type: none"> Initial design to fine tuning.

Parametric-based optimization on the cloud

- Using a feature-based, fully parametric CAD application gives the designer incredible level of control over the solid model.
- But the problem with most CAD apps, is that they do not work in Linux and they do not take input parameters using scripting language.
- In general, they are not easy to introduce in an optimization loop.
- To overcome this problem, we can use Onshape (www.onshape.com).
 - Full cloud based professional 3D CAD system.
 - Fully collaborative and simultaneous real time editing.
 - It runs on any device with a working web browser.
 - Academic and public versions → Free.
 - Professional version → Monthly/annual subscription.
 - All versions share same capabilities.
 - RESTful API, so it can be scripted using python or nodeJS.

Parametric-based optimization on the cloud

- By using Onshape RESTful API, we are able to close our optimization loop using a fully parametric CAD system.



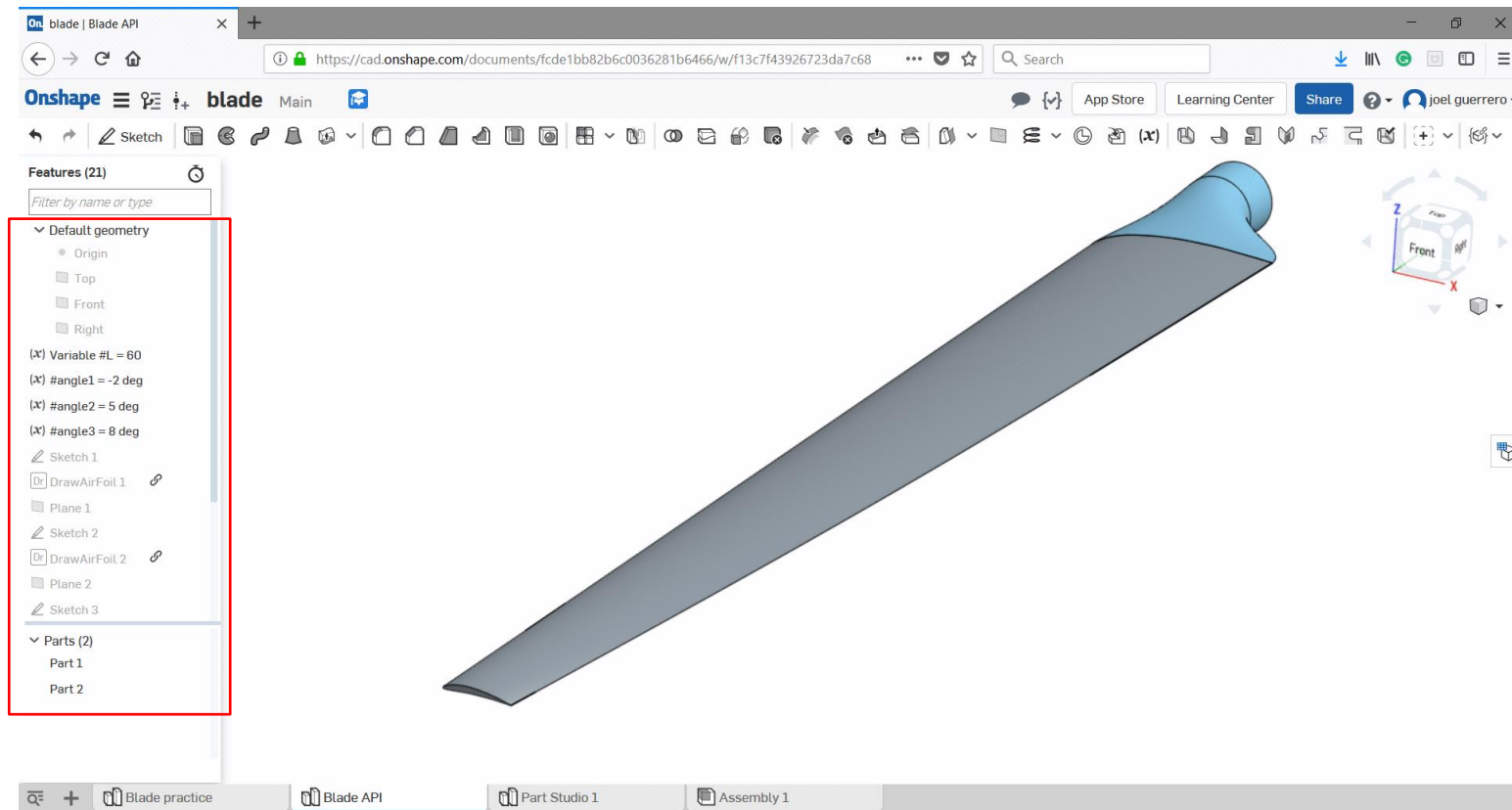
Parametric-based optimization on the cloud

- RESTful requests → POST, GET, PUT, DELETE
- Request a feature/document update or change.
- Get the request response.
- Download the new solid model in STL format or any CAD exchange format.

<http://www.wolfdynamics.com/training/opt/ani7.gif>



Any feature in the tree can be modified using Onshape RESTful API



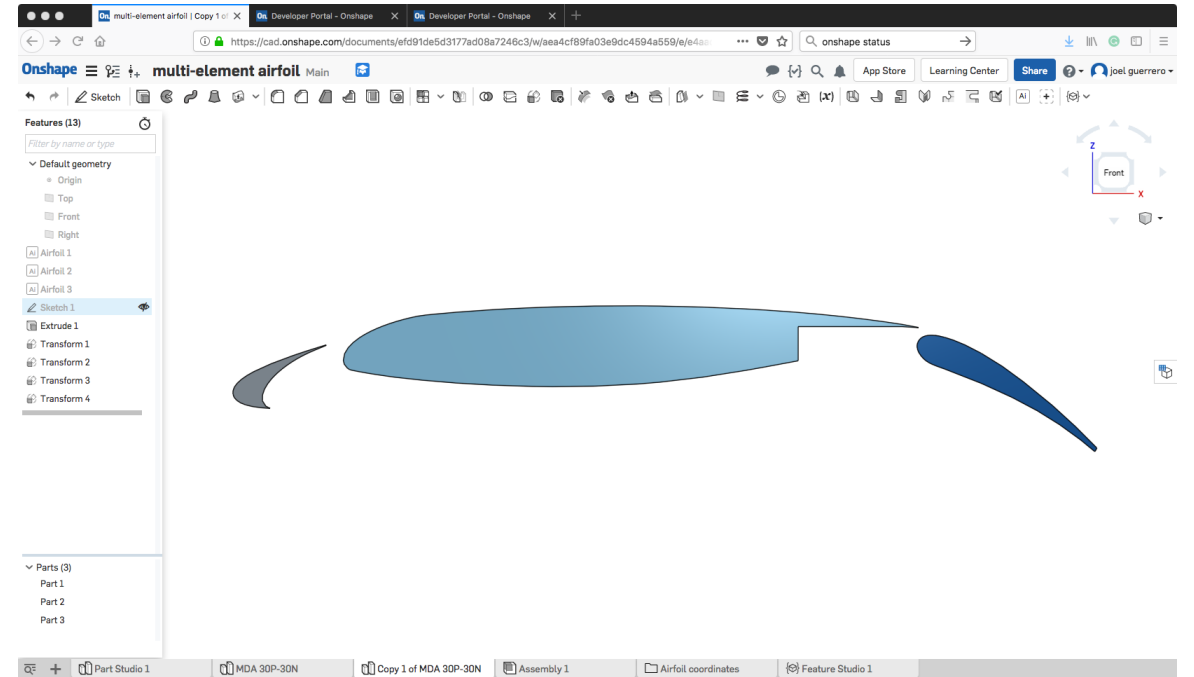
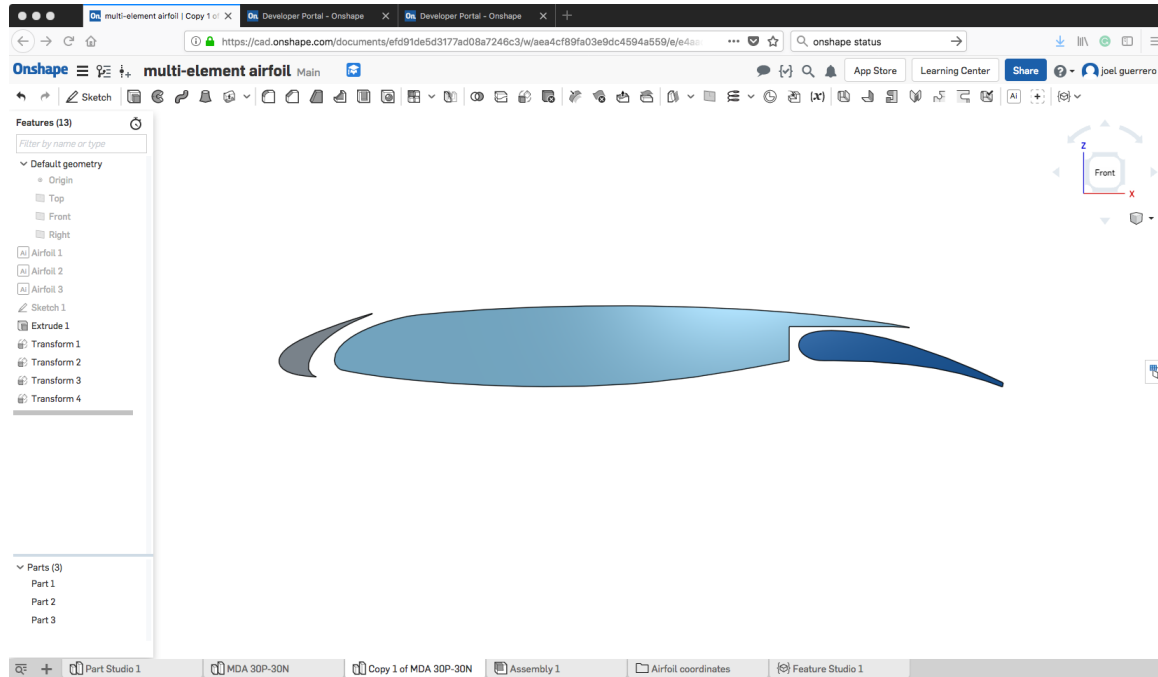
Oauth authentication



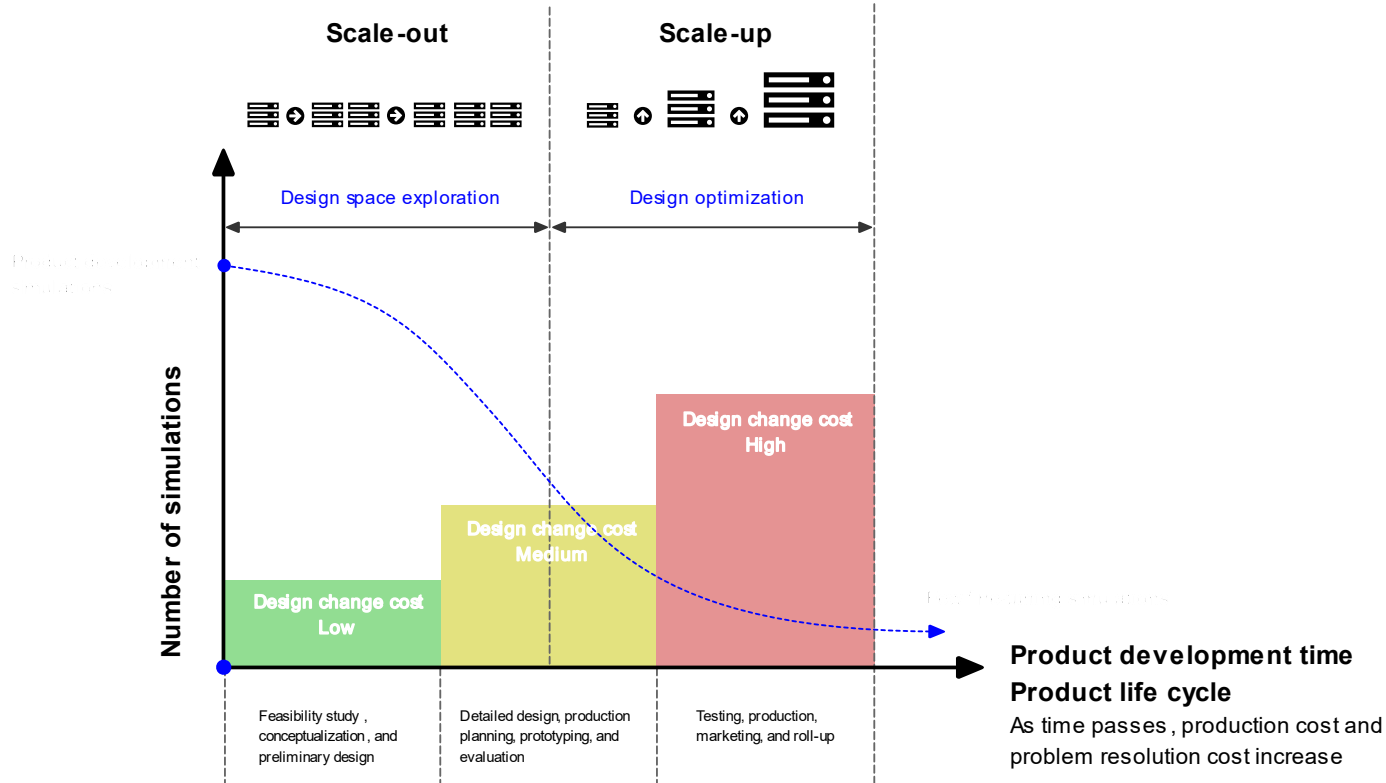
API – Python interface

Parametric-based optimization on the cloud

- By using a fully parametric CAD, things such as this high-lift wing can be easily parametrized.
- Doing such modifications using mesh morphing is not that easy and robust.
- But if you are still interested in working at the mesh level, a workaround can be the use of overset meshes.



Numerical simulations, product development, and the need of optimization



- To design innovative products in a cost-effective way and to improve their performance, the industry is relying more and more on numerical simulations, numerical optimization, exploratory data analysis* and business intelligence**.
- The benefits of simulating far outweighs the costs related to physical experiments and constructing prototypes.
- Simulate early, simulate often. Get it right the first time.
- Plan your optimization study and exploit your computational resources in an efficient way.

* Exploratory data analysis (EDA) is an approach to analyzing data sets to summarize their main characteristics, often with visual methods.

** Business intelligence (BI) comprises the strategies and technologies used by enterprises for the data analysis of business information. BI technologies provide historical, current and predictive views of business operations.

Final remarks

- Can optimization methods guarantee the existence and uniqueness of a global (or a local) optimum solution?
 - **The short answer is no.**
- Optimization is very subjective.
- So, for a designer or engineer an optimal value found by the optimizer might not be a practical solution.
- It is also not uncommon to have multiple solutions, as in the case of multi-objective optimization.
- And many times, we need to optimize concepts a little bit more abstract, such as, customer satisfaction, manufacturing process, packing factor, form factor, idle time, cost reduction. And formulating these kind of problems is not very easy.
- Also, ill-conditioned problems can result in poor convergence.

Final remarks

- And to make matters worst, in engineering design we often deal with multi-disciplinary optimization (MDO).
 - For example, improving the aerodynamic performance of an airplane (aerodynamics group), can result in larger weight of the airplane due to larger aerodynamic loads (structural group), and this might reduce the payload and the handling qualities of the airplane (flight performance group), and therefore the revenues (sales department).
- At the end of the day, the current design iteration should be an improvement of the previous iterations.
- And we find a better solution by using any of the methods we just described.
- We aim at using sound approach instead of the what-if approach (guessing).

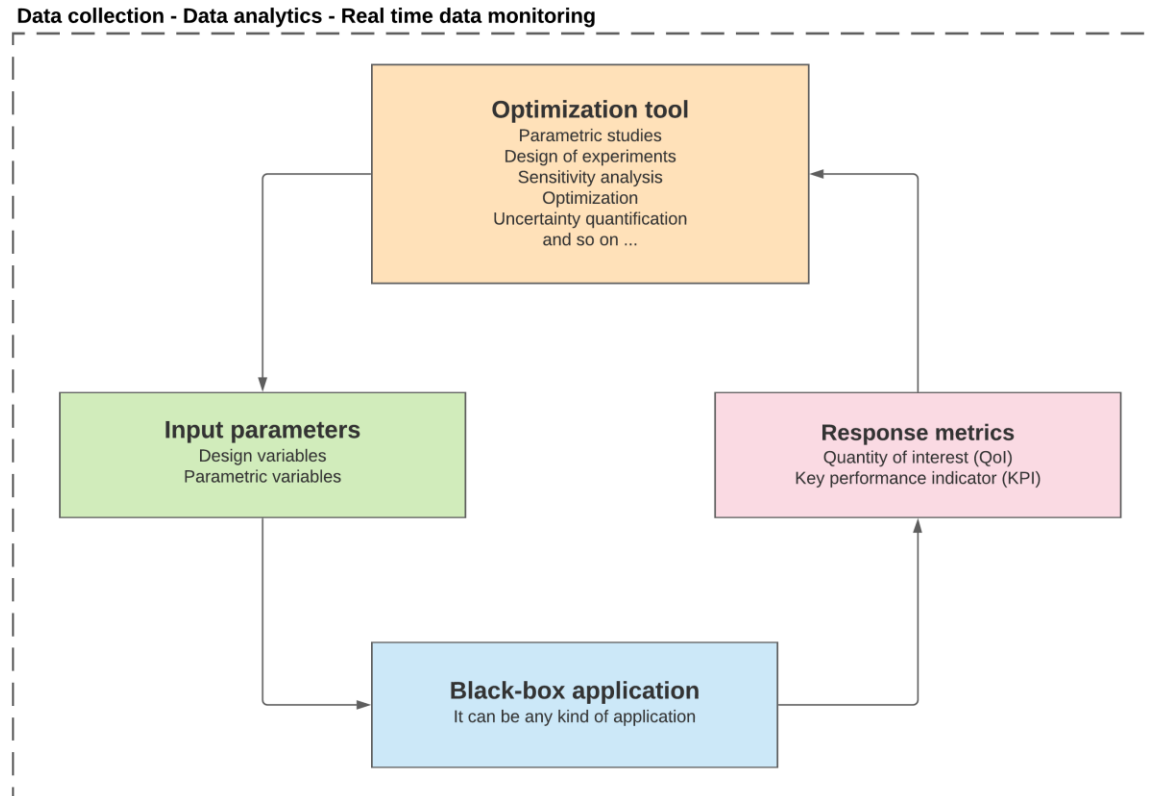
Final remarks

- Finally, have in mind that,
 - Optimization is a slow converging, meticulous and thoughtful process that requires careful planning, fault tolerant loops, and real-time data monitoring and analysis.
 - Do not expect fast outcomes leading to miraculous solutions.

2. CFD optimization loops – The big picture

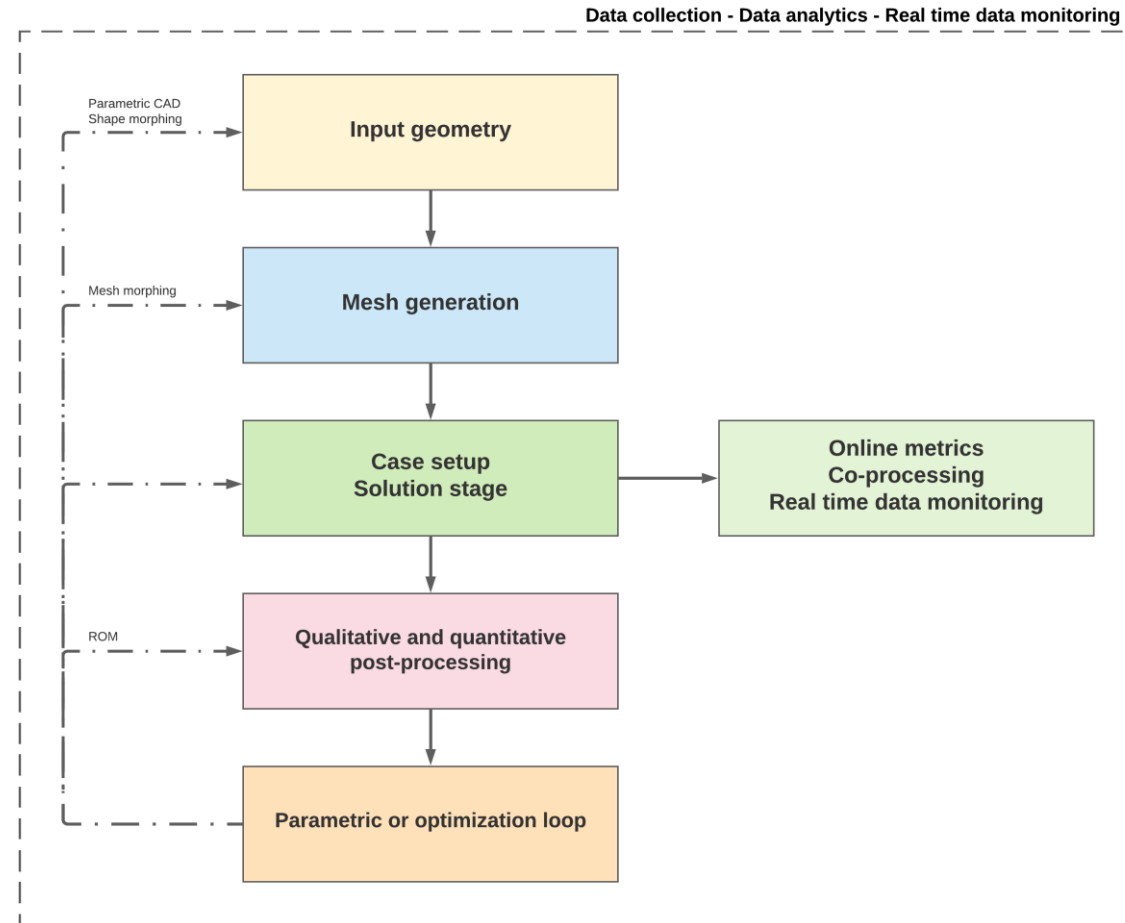
CFD optimization loop – The big picture

- A general optimization loop can be used in any field.
- The input parameters can be any kind of design variable (numerical, categorical, text, an image, and so on).
- The quantity of interest (QOI) can be any kind quantitative or qualitative data (numerical, categorical, text, an image, and so on).
- The black-box application can be any kind of application and preferably it should be able to interact via the command line interface or using parametric input files.



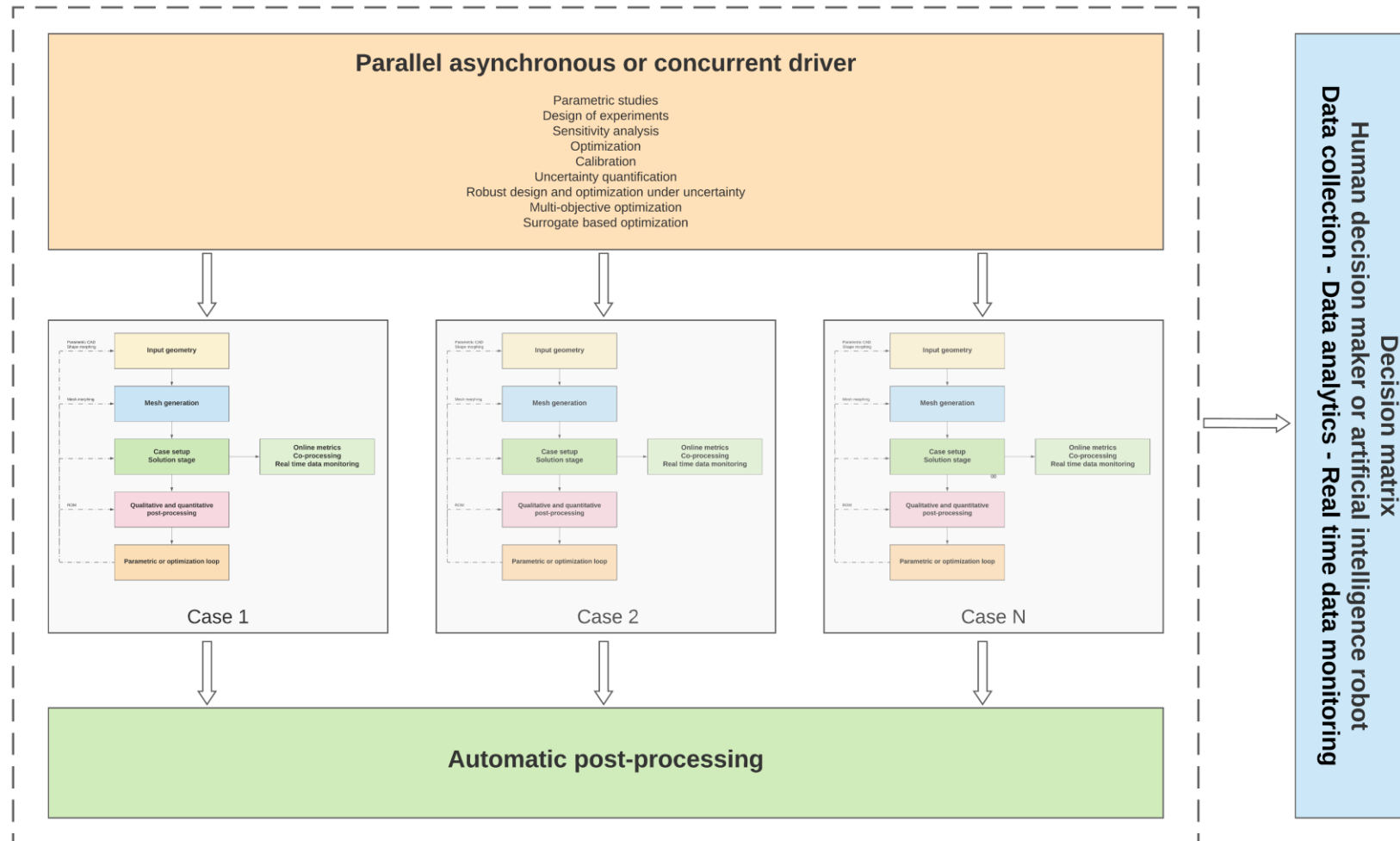
CFD optimization loop – The big picture

- Typical vertical approach when conducting CFD studies and launching one simulation at a time.
- In this vertical approach the engineering design loop can iterate back to any of the previous steps.
- This single case can be run in parallel.



CFD optimization loop – The big picture

- Concurrent engineering design loop for CFD studies.
- Here we use many processors to solve many problems at the same time.
- At the same time, we solve each problem using many processors.



CFD optimization loop – The big picture

- From the point of view of code coupling, there are two types of optimization loops:
 - **Loosely coupled approach**
 - Where all the applications interact via shell scripting or the command line interface.
 - Usually, the applications belong to different software developers and do not share common data exchange formats.
 - Not very user friendly.
 - **Strongly coupled approach**
 - where all applications interact using a single interface, graphical or command line based.
 - Often, the applications belong to the same software developer, and they share a common data exchange format.
 - User friendly (to some extension).

CFD optimization loop – The big picture

- The **strongly coupled approach** is commonly found when using commercial applications, such as, Ansys Fluids, StarCCM, VisualDOC, CAESES.
- The **loosely coupled approach**, is found when using open-source applications or applications that do not belong to the same software developer or are not meant to interact with other applications.
- From the performance point of view, the **strongly coupled approach** performs better due to the tight integration.
- But this does not mean that the **loosely coupled approach** is much worse.
 - By using good practices, the user can get an acceptable performance when integrating different applications.
 - It is also more flexible, because it allows user to use applications that do not belong to the same software developer, which may be a limitation when using a GUI based **strongly coupled approach**.
- Hereafter, we will work with a **loosely coupled approach**.
 - We will propose an engineering design loop using DAKOTA as optimizer and code coupling application.
 - Then, by using the command line interface and shell scripting, we will integrate different applications.

CFD optimization loop – The big picture

- A few open-source applications that can be used with a loosely coupled engineering design loop.

- **Code coupling/Optimizer:**

- DAKOTA – <https://dakota.sandia.gov/>
- RAVEN – <https://raven.inl.gov/>
- OpenMDAO – <https://openmdao.org/>

- **Concurrent computations scheduler:**

- DAKOTA – <https://dakota.sandia.gov/>
- Python
- Bash shell and GNU parallel

- **Parametric CAD:**

- Onshape (API) – <https://onshape-public.github.io/>
- FreeCAD – <https://www.freecadweb.org/>
- SALOME – <https://www.salome-platform.org/>
- OpenSCAD – <https://www.openscad.org/>

- **Meshing tools:**

- GMSH – <https://gmsh.info/>
- TETGEN – <http://wias-berlin.de/software/tetgen/>
- SALOME – <https://www.salome-platform.org/>
- pyhyp – <https://github.com/mdolab/pyhyp>
- MeshKit – <https://sigma.mcs.anl.gov/meshkit-library/>

- **Mesh morphing tools**

- MESQUITE – <https://trilinos.github.io/mesquite.html>
- PyGeM – <https://mathlab.sissa.it/pygem>
- idwarp – <https://github.com/mdolab/idwarp>
- pygeo – <https://github.com/mdolab/pygeo>

- **Black-box solver:**

- OpenFOAM – <https://openfoam.org/>
- SU2 – <https://su2code.github.io/>
- FLUBIO-PETSC – <https://flubiopetsc.github.io/flubiopetsc/>

- **Quantitative and qualitative post-processing:**

- Python, paraview, JavaScript, VTK

- **Real time data monitoring:**

- Python, R, JavaScript, BASH

- **Exploration and exploitation of design space:**

- Python, R, BASH

- **Additional automation scripting:**

- Python, BASH

CFD optimization loop – The big picture

- The automatic loop should cover the whole workflow of a CFD simulation:

Solid modeling → Meshing → Case setup → Simulations and monitoring → Post-processing

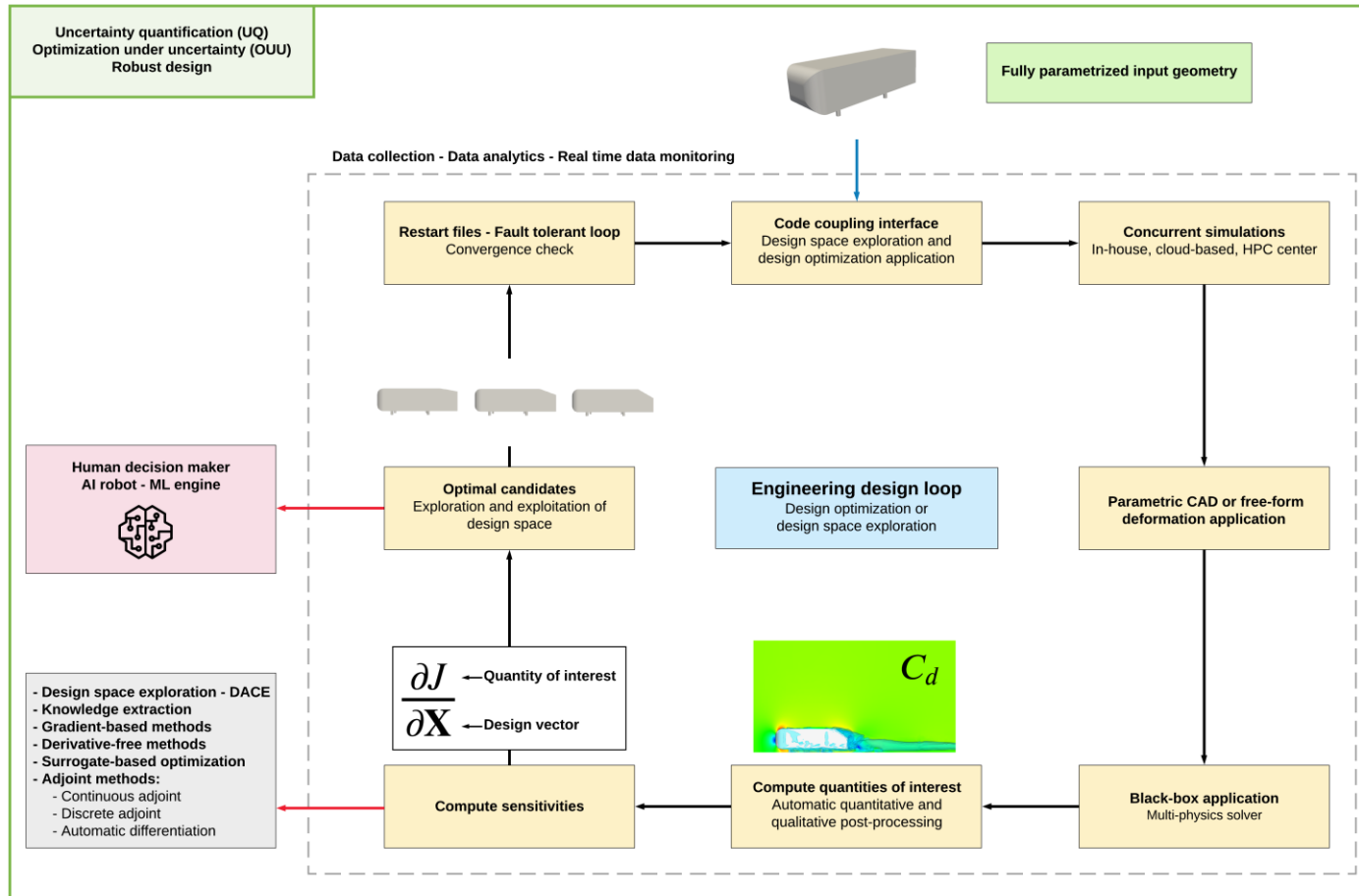
- We will illustrate an automatic loop for simple simulations with many applications interacting.
- But have in mind that the proposed framework can be easily extended to deal with complex scenarios
- And as we are dealing with CFD, the proposed framework can cover different fields in science and engineering (aerospace, automotive, HVAC, AEC, medical devices, thermal management, naval, and so on).
- For the cases that we will be presenting, the simulations are run using a pre-specified level of accuracy and iterative marching (which is not bad).
- However, by using data and metadata (data-of-data) to compute basic descriptive statistics and by leveraging a few concepts of SL/ML, the design loop can freely iterate until it reaches an acceptable level of convergence.

CFD optimization loop – The big picture

- A few comments on the automatic loop or optimization framework to be proposed:
 - The framework is automatic and to some extent fault tolerant.
 - But in the case of fatal failure, the user can restart from the latest stable solution.
 - In the case of anomalies while the loop is running, the input parameters can be changed on-the-fly to stabilize the solution, this can be done automatically (a lot SL/ML involved) or manually.
 - To achieve this, a lot of things need to be monitored.
 - Therefore, it is important to monitor all the QOIs and KPIs real-time.
 - Every single modification is recorded and reported to the user.
 - The bottleneck is the meshing stage (at least when using OpenFOAM tools, and in particular snappyHexMesh).
 - In case of meshing failure or bad quality meshes, the domain is remeshed using more robust parameters (which will increase the meshing time and mesh size).
 - If the mesh issues cannot be repaired in an automatic way, the user must fix the problems manually, which is not desirable.

CFD optimization loop – The big picture

- For example, a loosely coupled approach for shape optimization in CFD can look like the following one.
- To conduct different tasks in the optimization loop, we use different tools that can interact between each another using shell scripting or the command line interface.
- Also, all the proposed applications are open-source.



- **Code coupling/Optimizer:**
DAKOTA
- **Concurrent computations scheduler:**
DAKOTA
- **Parametric CAD:**
Onshape (API)
- **Black-box solver:**
OpenFOAM
- **Quantitative and qualitative post-processing:**
Python, paraview, JavaScript
- **Real time data monitoring:**
Python, R, BASH
- **Exploration and exploitation of design space:**
Python, R, BASH
- **Additional automation scripting:**
Python, BASH

CFD optimization loop – Automation, parametrization, and concurrency

- If you are planning to do optimization studies, things need to be automatic and parametric.
- It is also important to take advantage of concurrency, that is, running many simulations at the same time.
- While iterating to the optimal solution, do not forget to use real-time data analytics and exploratory data analysis to study the data gathered.
- Sometimes there might be a hidden story in this data.
- Avoid to iterate in your optimization loop manually, it is too slow and prone to errors.
- As stated in the NASA contractor report “CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences”,

“ A single engineer/scientist must be able to conceive, create, analyze, and interpret a large ensemble of related simulations in a time-critical period (e.g., 24 hours), without individually managing each simulation, to a pre-specified level of accuracy. ”

3. DAKOTA overview

DAKOTA overview

- DAKOTA stands for **D**esign and **A**nalysis tool**K**it for **O**ptimization and **T**erascale **A**pplications.
- DAKOTA is a general-purpose software toolkit for performing optimization, uncertainty quantification, parameter estimation, design of experiments, and sensitivity analysis on high performance computers.
 - DAKOTA is developed and supported by U.S. Sandia National Labs.
 - It is well documented and comes with many tutorials.
 - Support via a dedicated mailing list.
- You can download DAKOTA toolkit at the following link: <http://dakota.sandia.gov/>
- Official releases and nightly stable releases are freely available worldwide via GNU GPL.
- Current version (as of June 2021): 6.14



DAKOTA overview

- DAKOTA capabilities:
 - Parameter Studies (PS).
 - Design of Experiments (DOE) – Design and Analysis of Computer Experiments (DACE) .
 - Sensitivity Analysis (SA).
 - Uncertainty Quantification (UQ).
 - Optimization (OPT) via Gradient-based methods, and derivative-free local and global methods.
 - Surrogate based optimization (SBO).
 - Calibration (CAL) or data fitting – Parameter estimation or Nonlinear Least Squares Capabilities.
 - Generic interface to black box solvers.
 - Scalable parallel computations from desktop to clusters.
 - Asynchronous evaluations.
 - Simulation failure capturing.
 - Restart capabilities.
 - Matlab, scilab, AMPL, Python interface.
 - Time-tested and advanced research algorithms to address challenging science and engineering simulations.

DAKOTA overview

- Why DAKOTA? Why not matlab, scilab, octave, Java, Python, R, or any other optimization framework?
- DAKOTA does pretty much what any other optimizer does.
- What differentiates DAKOTA from any other optimization tool is,
 - Generic interface to black box solvers.
 - Scalable parallel computations, from desktop to clusters to the cloud.
 - Extensively validated.
 - Fully scriptable.
 - Simulation failure capturing.
 - Restart capabilities.
 - Parallel asynchronous or concurrent evaluations.
 - Can be linked to third-party optimization libraries.
 - No license fee.
 - Open-source.

DAKOTA overview

- DAKOTA uses a single input file to orchestrate the optimization loop.
- This input file is human readable (ASCII format) and has the extension ***.in** (the extension is superfluous in UNI/Linux like OS).
- In this input file the user formulates the problem, that is:
 - Method to use, variables, and responses.
 - Additionally, the user can define the interface to the black box solver, the level of parallelism, and the general settings such as tabular output (the environment).
- If you misspell something or use a keyword that does not exist in the input file, DAKOTA will list the available options.
- Also, if you forget a compulsory entry in the input file, DAKOTA will complain and will ask you for that value.
- Optional entries will use the default values.
- Refer to the documentation for more information about the compulsory and optional entries of each method.

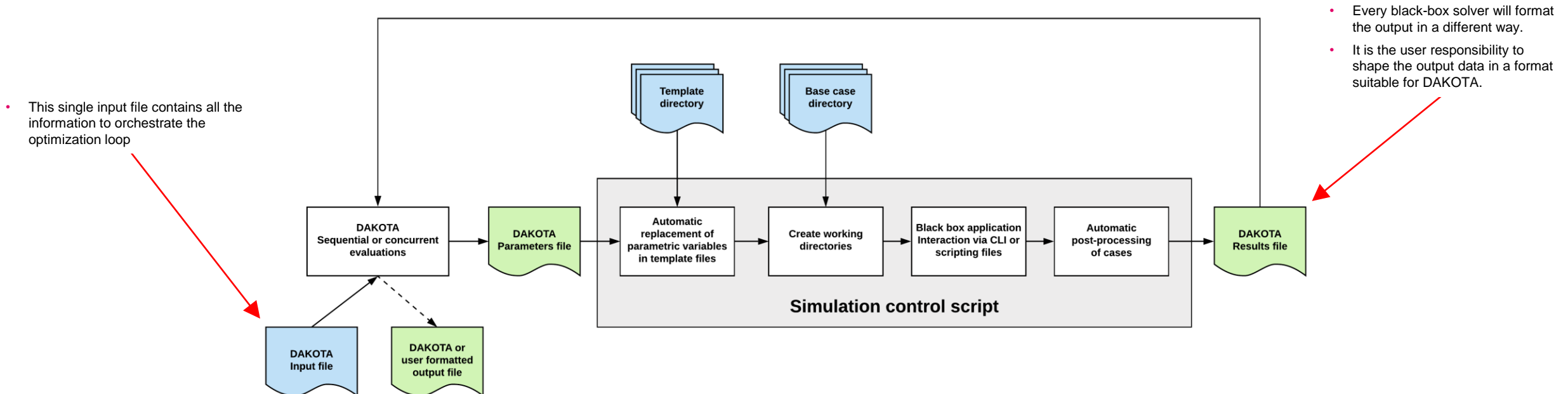
DAKOTA overview

- DAKOTA Input file (e.g., input.in).

Define algorithm	<pre>environment tabular_data tabular_data_file = 'output.dat'</pre>	environment (required): specify general settings such as tabular output. It also identifies the top-level method.
	<pre>method max_iterations = 100 convergence_tolerance = 1e-4 conmin_frcg</pre>	method (required): specifies the method (DO or DSE) and specific settings
Define problem (inputs and outputs) Set interface	<pre>model single</pre>	model (optional): a model provides a logical unit for determining how a set of variables is mapped into a set of responses. The model allows one to specify a single interface or to manage more sophisticated mappings involving surrogates or nested iterations. Default value is single.
	<pre>variables continuous_design = 2 initial_point -1.2 1.0 lower_bounds -2.0 -2.0 upper_bounds 2.0 2.0 descriptors 'x1' 'x2'</pre>	variables (required): parameters input to the simulation; these are the design variables.
	<pre>Interface fork analysis_driver = 'simulator_script' asynchronous</pre>	interface (required): map from variables to responses; control parallelism
	<pre>responses objective_functions = 1 numerical_gradients no_hessians</pre>	responses (required): model output(s) to be studied, these are the response metrics.

DAKOTA overview

- Workflow for data exchange between Dakota and a black-box application.
 - The white rectangles denote process blocks.
 - The light-shaded blue document symbols denote unchanging sets of files.
 - The light-shaded green document symbols indicate files that change with each set of design parameters generated by Dakota or after the end of the evaluation of the QOI.
 - The light-shaded grey area denotes the domain of the control script that automatically prepares the case; including, automatic formatting of input and output files, organization of the data generated, and executing the application in serial or parallel.



DAKOTA overview

- A few optimization methods and design exploration methods available in DAKOTA.

Gradient-based Optimization:

- **CONMIN:** frcg, mfd
- **OPT++:** cg, Newton, quasi-Newton
- **DOT:** frcg, bfgs, mmfd, slp, sqp (external package – commercial)
- **NPSOL:** sqp (external package – commercial)
- **NLPQLP:** sqp (external package – commercial)

Derivative-free Optimization:

- **COLINY:** PS, EA, Solis-Wets, COBYLA, DIRECT
- **JEGA:** MOGA, SOGA
- **EGO:** efficient global optimization via Gaussian Process models
- **NCSU:** DIRECT
- **OPT++:** PDS (Parallel Direct Search, simplex based method)
- **NOMAD:** mesh_adaptive_search

Parameter studies: vector, list, centered, grid, multidimensional, user defined

Design of experiments:

- **DDACE:** LHS, MC, grid, OA, OA_LHS, CCD, BB
- **FSUDace:** CVT, Halton, Hammersley
- **PSUADE:** MOAT
- **Sampling:** LHS, MC, Incr. LHS, IS/AIS/MMAIS

Multi-objective optimization, pareto, hybrid, multi-start, surrogate-based optimization (local and global), uncertainty quantification.

DAKOTA overview

- DAKOTA comes with extensive documentation and tutorials.
- You can access or download DAKOTA's documentation from the following link:

<https://dakota.sandia.gov/content/manuals>


Documentation tab

Sandia National Laboratories

Search Sandia.gov | All Sandia Websites

Home Download Documentation Community About Login (SNL Only)

Get Started Manuals Publications Release Notes Screencasts

 **DAKOTA**
Explore and predict with confidence.

Manuals

Dakota provides the following manuals:

- **User's Manual** — The most comprehensive manual, containing overall information on Dakota, including its purpose and capabilities, a tutorial, interfacing guide, examples, and more.
- **Reference Manual** — A detailed input specification guide organized in the same hierarchical format as the Dakota input file
- **Developer's Manual** — Dakota design principles, services, best practices for team members, and development resources.
- **Theory Manual** — This manual accompanies the Users' Manual and provides more mathematical detail and deeper discussion of Dakota methods.
- **GUI User Manual** — A manual for the Dakota GUI, which doubles as a step-by-step tutorial to help users get studies up and running in the GUI.

Notes:

- For the Reference and Developer's Manual *we strongly recommend using the HTML versions* (online or downloaded as tar.gz). The PDFs are provided for reference / archival purposes, but are far less usable.
- *Major numbered releases M.m are for production use.* Stable releases, indicated by a '+' if available, are developmental which should be used with greater caution.

Version 6.14

- User's Manual — pdf
- Reference Manual — html, tar.gz, (pdf only if needed)
- Developer's Manual — html, tar.gz, (pdf only if needed)
- Theory Manual — pdf
- GUI Manual — html

Reference documentation of all available methods

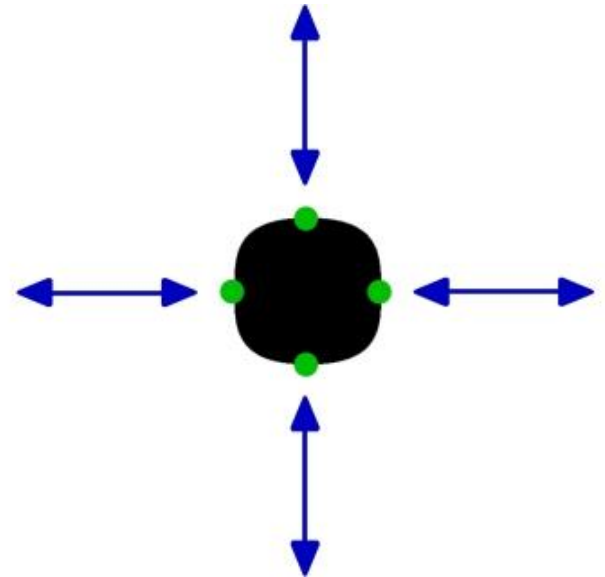
4. CFD optimization loops in action – Some examples

CFD optimization sample cases:
Shape optimization of a blunt body

CFD optimization sample cases

- Blunt body shape optimization:

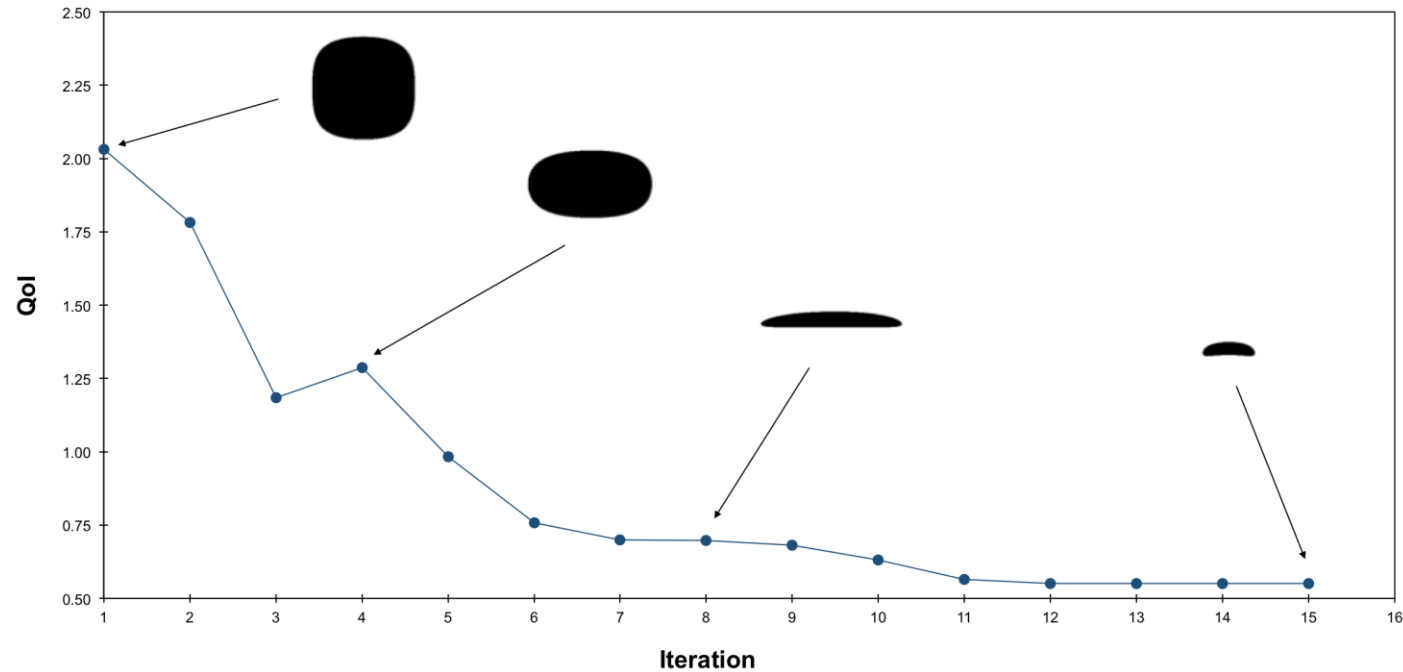
- In this case we aim at optimizing the shape of a blunt body.
- The Reynolds number is 1000 and depending on the shape of the body, the flow might exhibit a strong unsteady behavior.
- The goal is to minimize the drag coefficient.
- The body is fully parametrized using Bezier curves with four control points, four linear constraints and one non-linear constraint.
- We will use all the optimization methods: monolithic gradient-based, design space exploration, SBO, and adjoint.
- Recall that gradient-based optimization relies on a well formulated problem: initial point, step-size, tolerances, goal, linear and nonlinear constraints, how to compute the gradients (forward, differences, central differences, complex step), hessian information (if needed by the method), type of design variables (integers, floats, cardinal) and so on.



CFD optimization sample cases

- Gradient-based optimization:

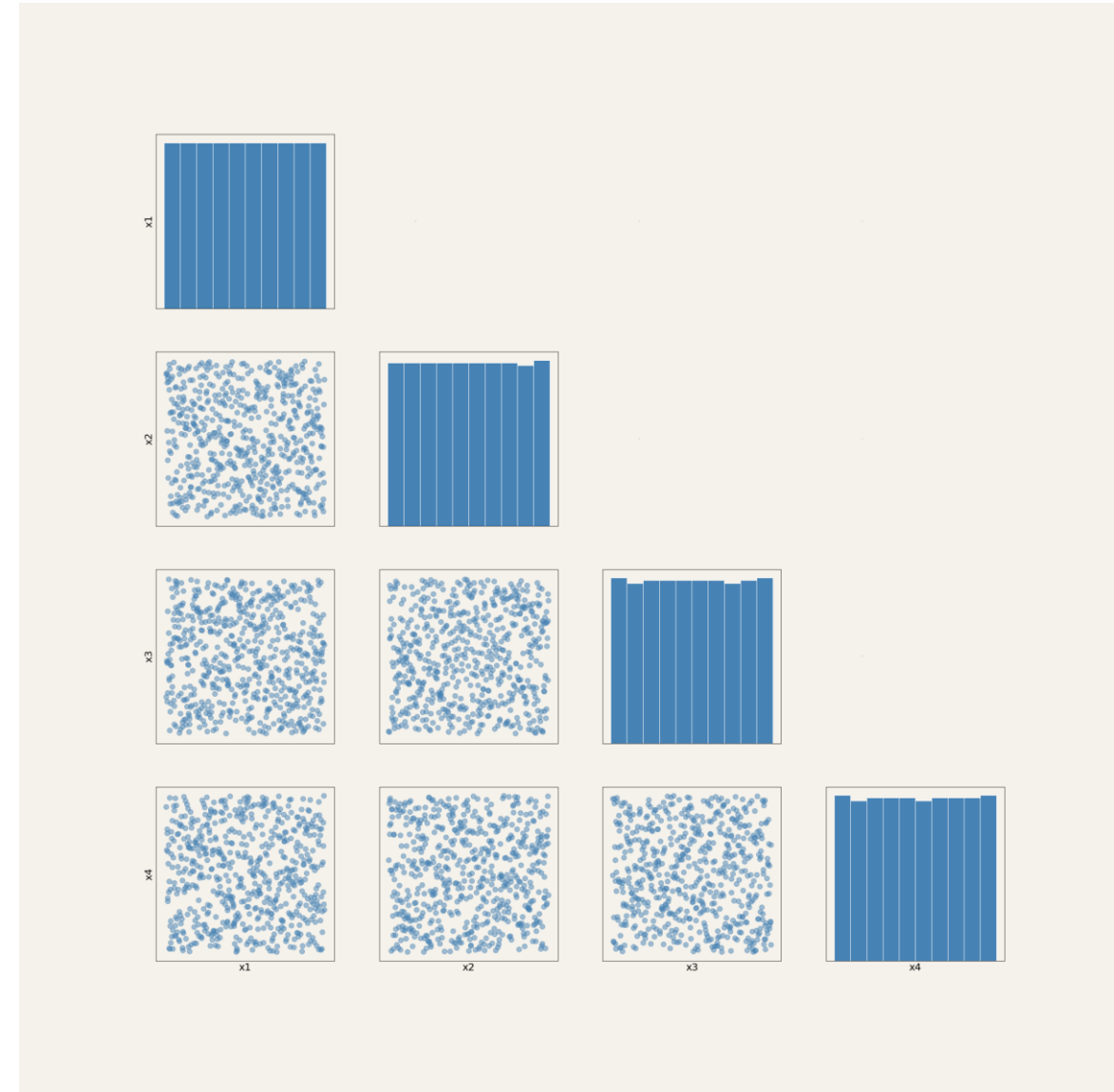
- Using a gradient-based optimization method, namely, the Method of Feasible Directions (MFD) the loop converged to the optimal shape in 15 function evaluations
- The gradients were evaluated using forward differences, and five gradient evaluations were required (one for each design variable).
- This problem converged in about 650 seconds.



CFD optimization sample cases

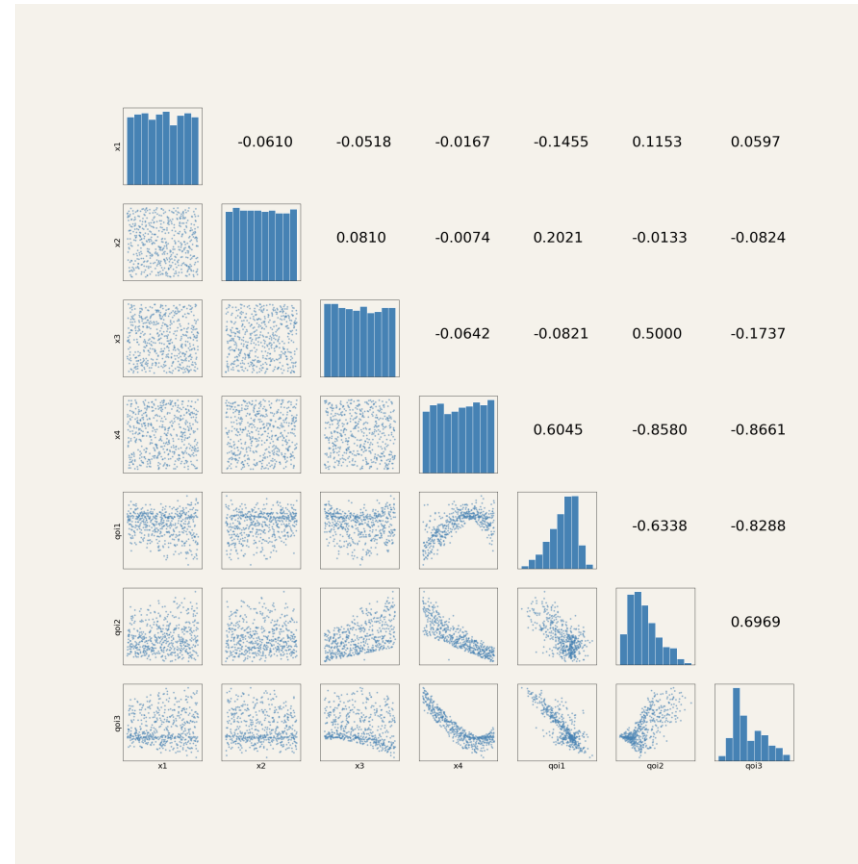
- Design space exploration (DSE):

- DSE does not formerly converge to the optimal value, it simply explores the design space uniformly.
- DSE is about knowledge extraction, finding patterns and correlations, and detecting anomalies.
- The output of DSE studies can be used to construct surrogate models.
- In this case we used a LHS sampling (latin-hypercube), with 600 samples.
- This problem converged in about 2100 seconds running 10 concurrent simulations.
- If we had more processors available, we would have converged faster (even faster than the gradient-based method).



CFD optimization sample cases

- With the information gathered, we can use exploratory data analysis, machine learning and statistical learning to analyze and visualize the information.



Scatter matrix plot of the design space exploration study – 600 experiments (high fidelity simulations)
This plot shows correlation, skewness, kurtosis, tendency and distribution of the data

CFD optimization sample cases

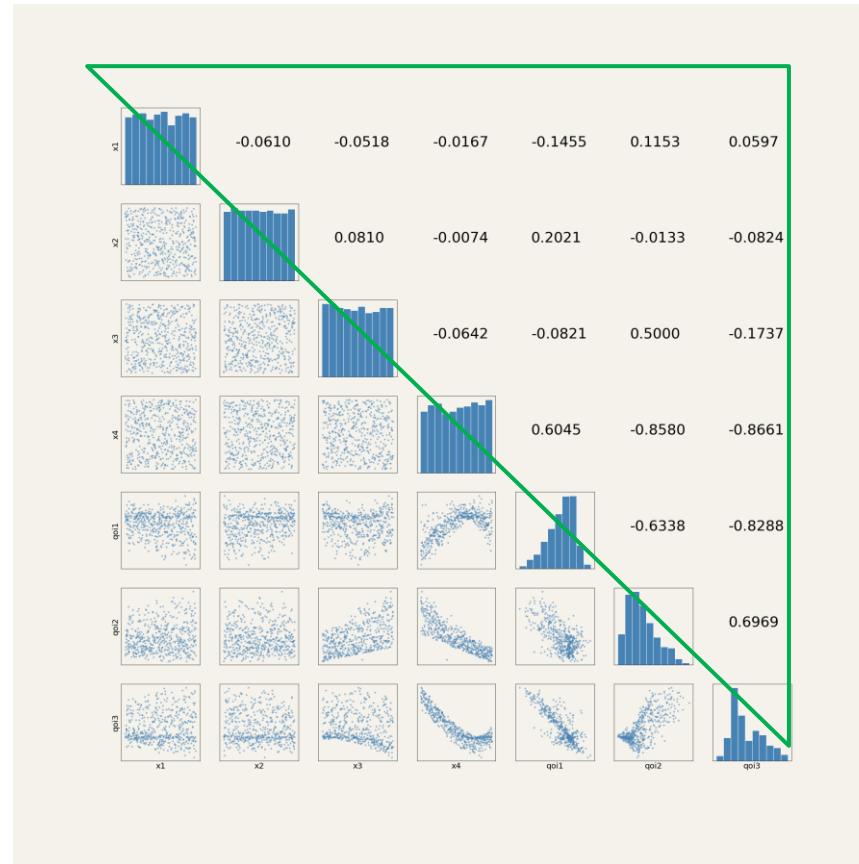
- With the information gathered, we can use exploratory data analysis, machine learning and statistical learning to analyze and visualize the information.



Scatter matrix plot of the design space exploration study – 600 experiments (high fidelity simulations)
Scatter plot of design variables distribution (sampling distribution in design space)

CFD optimization sample cases

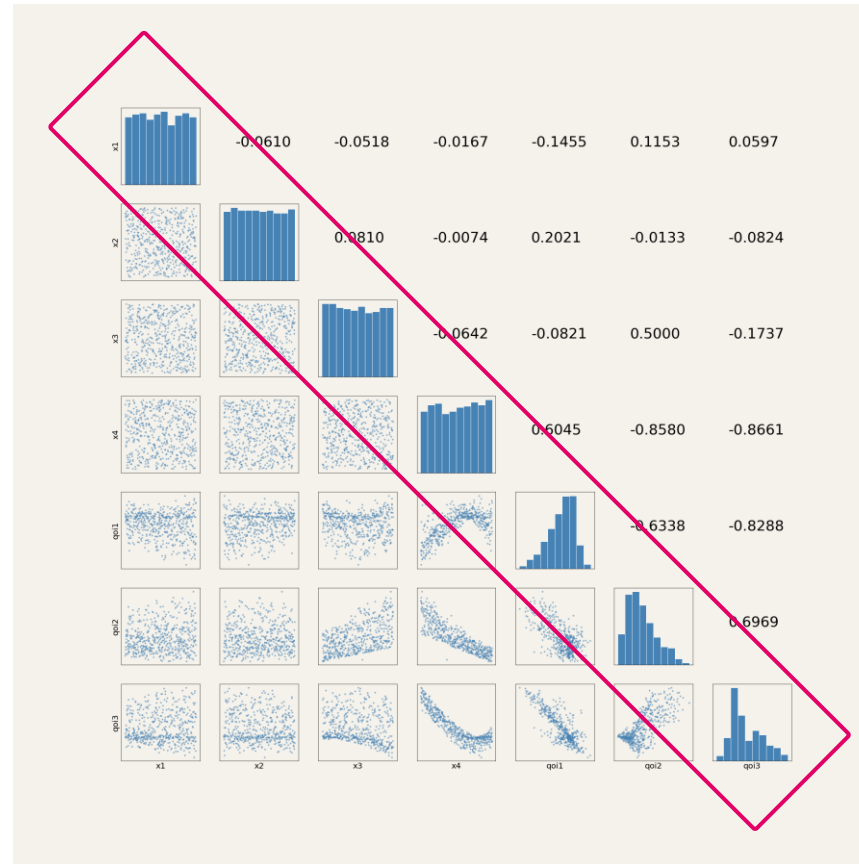
- With the information gathered, we can use exploratory data analysis, machine learning and statistical learning to analyze and visualize the information.



Scatter matrix plot of the design space exploration study – 600 experiments (high fidelity simulations)
Correlation matrix of design variables and objective functions

CFD optimization sample cases

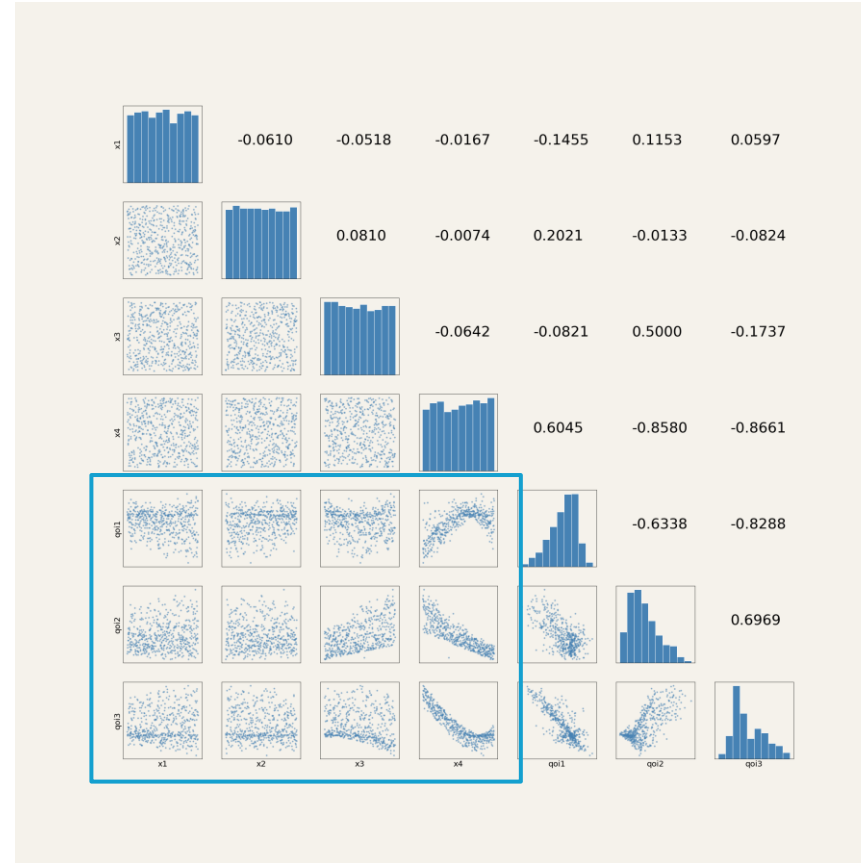
- With the information gathered, we can use exploratory data analysis, machine learning and statistical learning to analyze and visualize the information.



Scatter matrix plot of the design space exploration study – 600 experiments (high fidelity simulations)
Histograms of design variables and objective functions

CFD optimization sample cases

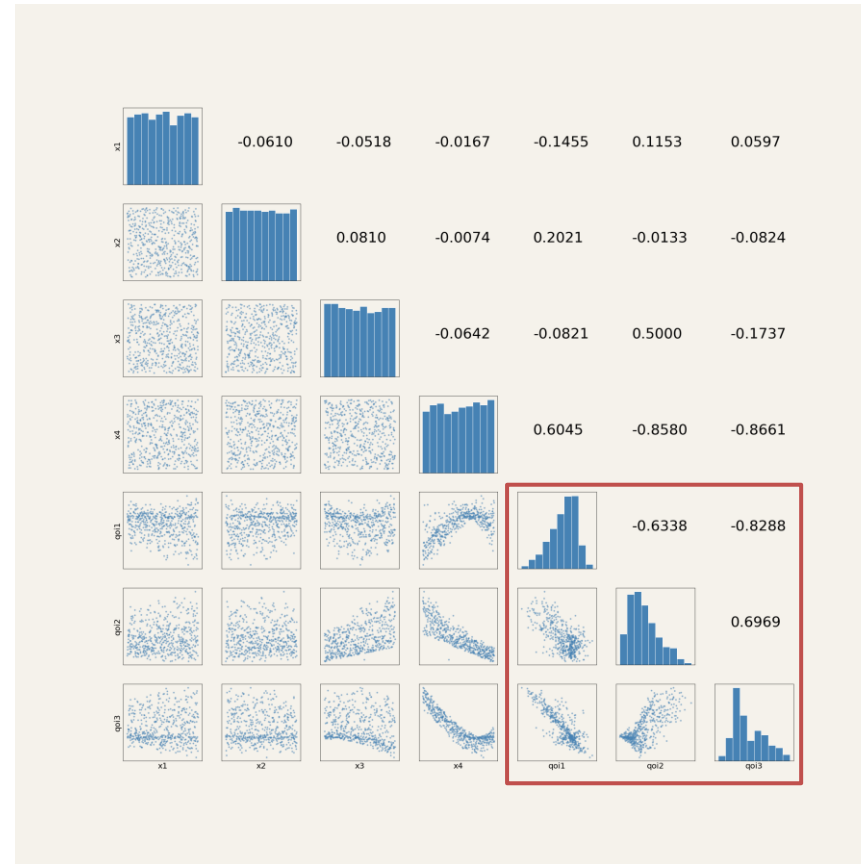
- With the information gathered, we can use exploratory data analysis, machine learning and statistical learning to analyze and visualize the information.



Scatter matrix plot of the design space exploration study – 600 experiments (high fidelity simulations)
Response of design space (scatter plot of design variables vs. objective functions)

CFD optimization sample cases

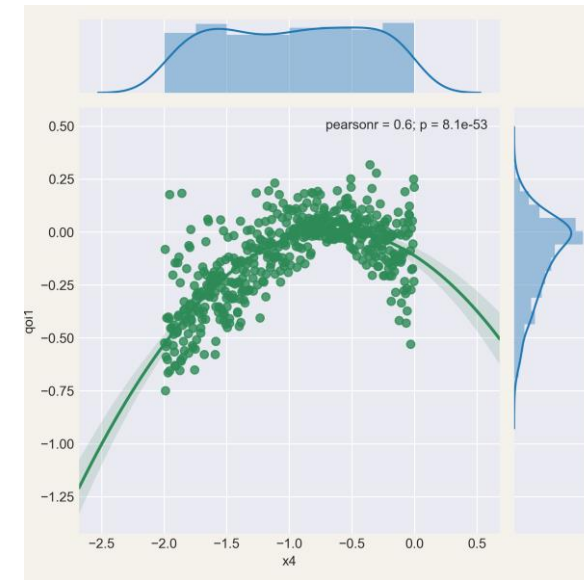
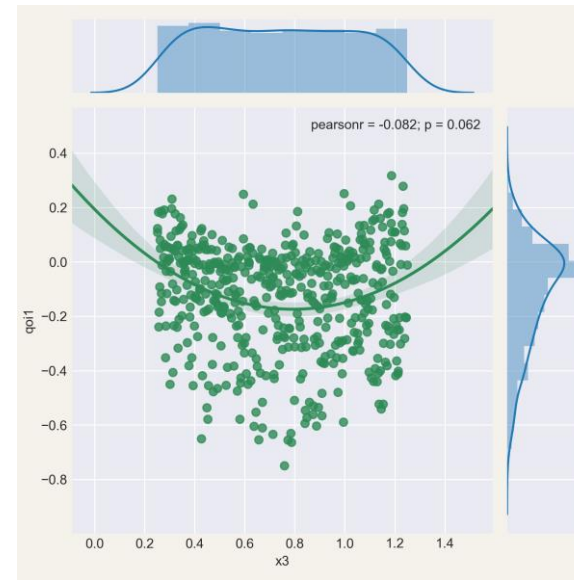
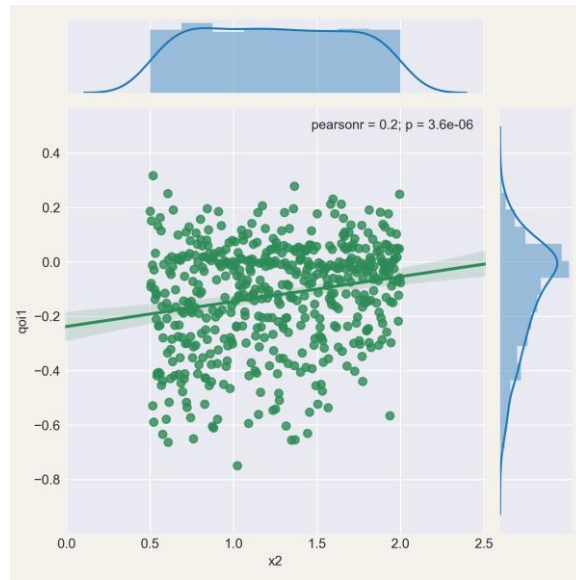
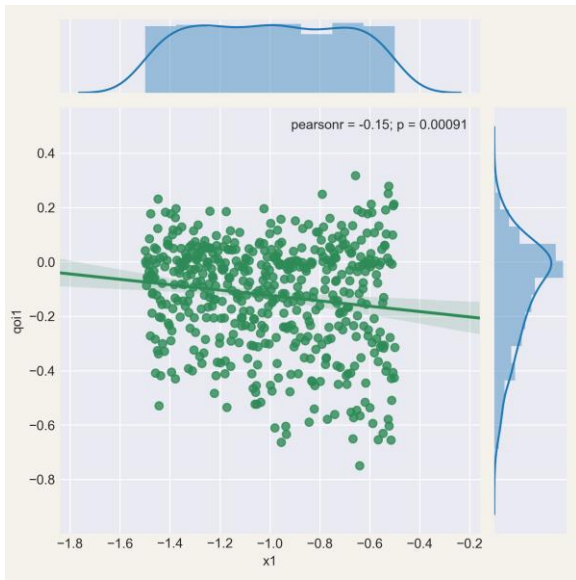
- With the information gathered, we can use exploratory data analysis, machine learning and statistical learning to analyze and visualize the information.



Scatter matrix plot of the design space exploration study – 600 experiments (high fidelity simulations)
Response or trade-off of objective functions

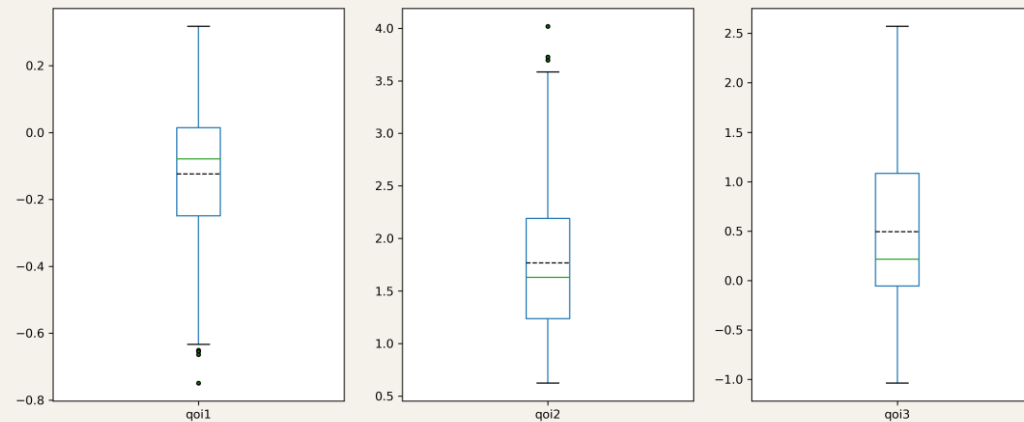
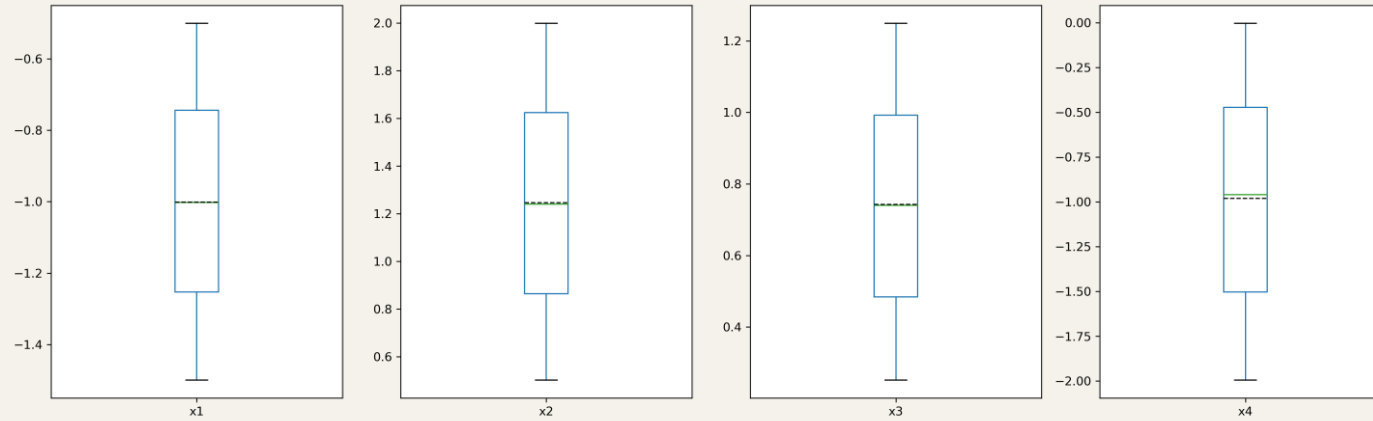
CFD optimization sample cases

- With DSE, besides visualizing the data, we can also construct regression models.
- In this case, we construct regression models (first and second order), to predict the response of the QOI with respect to the design variables.



CFD optimization sample cases

- We can use statistical analysis to detect outliers.
- In this case, we use box plot for outlier detection.

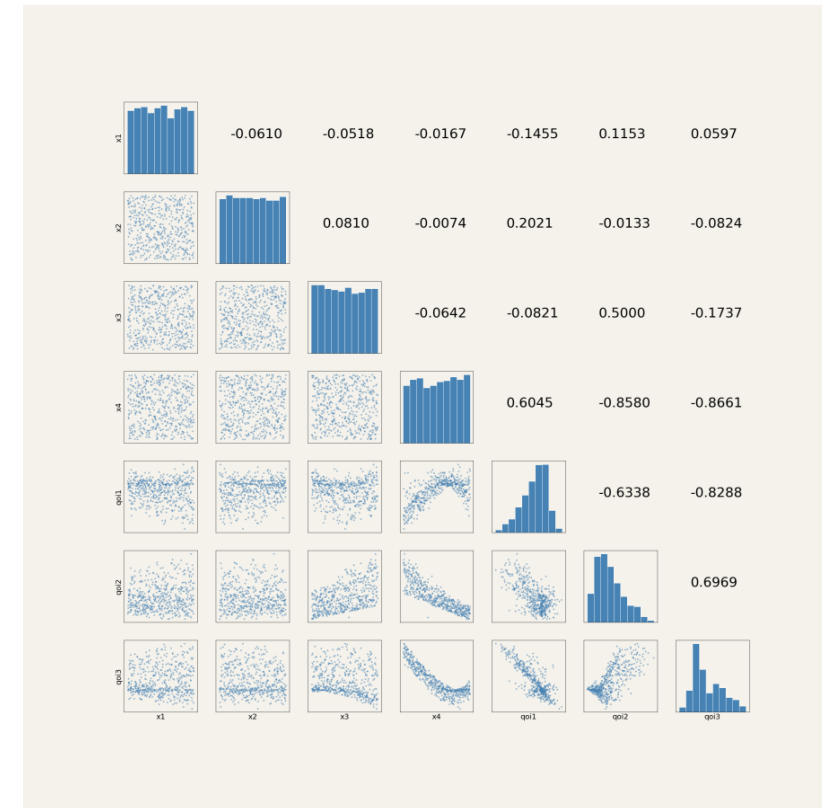


CFD optimization sample cases

- With DSE, we can also identify problems in the loop.
- A well-trained eye will see that there is a problem with the output of the QOIs, likely related with the parametrization.
- We do not need to rerun the whole experiment; we just need to clean the data.
- We can also extract the information of the cases that failed and rerun those cases.



Original data



Tidy data

CFD optimization sample cases

- In DSE, we are not vinculated to only one QOI.
- We can gather as much information as we like.
- We can even use existing data to compute new QOIs.
- This dataset, can be used to a surrogate model.

EVAL	X1	X2	X3	X4	QOI1	QOI2	QOI3
1	-1.236730769	1.981265363	0.4088770145	1.391058173	0.0510028	1.62929	0.830479
2	-0.7643307233	1.616373522	0.3299296905	-1.606818492	-0.239949	1.80283	1.58343
3	-1.432174929	1.587861754	1.166425805	-1.577506296	-0.156619	2.63285	0.784579
4	-1.079823003	0.7499768403	0.4472563114	-1.368531419	-0.179083	1.75741	1.00322
				...			
				N			

CFD optimization sample cases

- There are many techniques to construct metamodels, just to name a few:
 - Polynomial regression
 - Multivariate adaptive regression splines
 - Neural networks
 - Radial basis functions
 - Kriging – Gaussian process
- For example, using Kriging interpolation the predictor can be constructed as follows,

$$\hat{f}(\mathbf{x}) = g(\mathbf{x})^T \beta + r(\mathbf{x})^T R^{-1} (f - G\beta)$$

$g(\mathbf{x})$ vector of the constant global model or trend function

β vector of the least squares estimates of the trend function coefficients

$r(\mathbf{x})$ correlation vector with the data points (variogram). This term is constructed using a Gaussian correlation function.

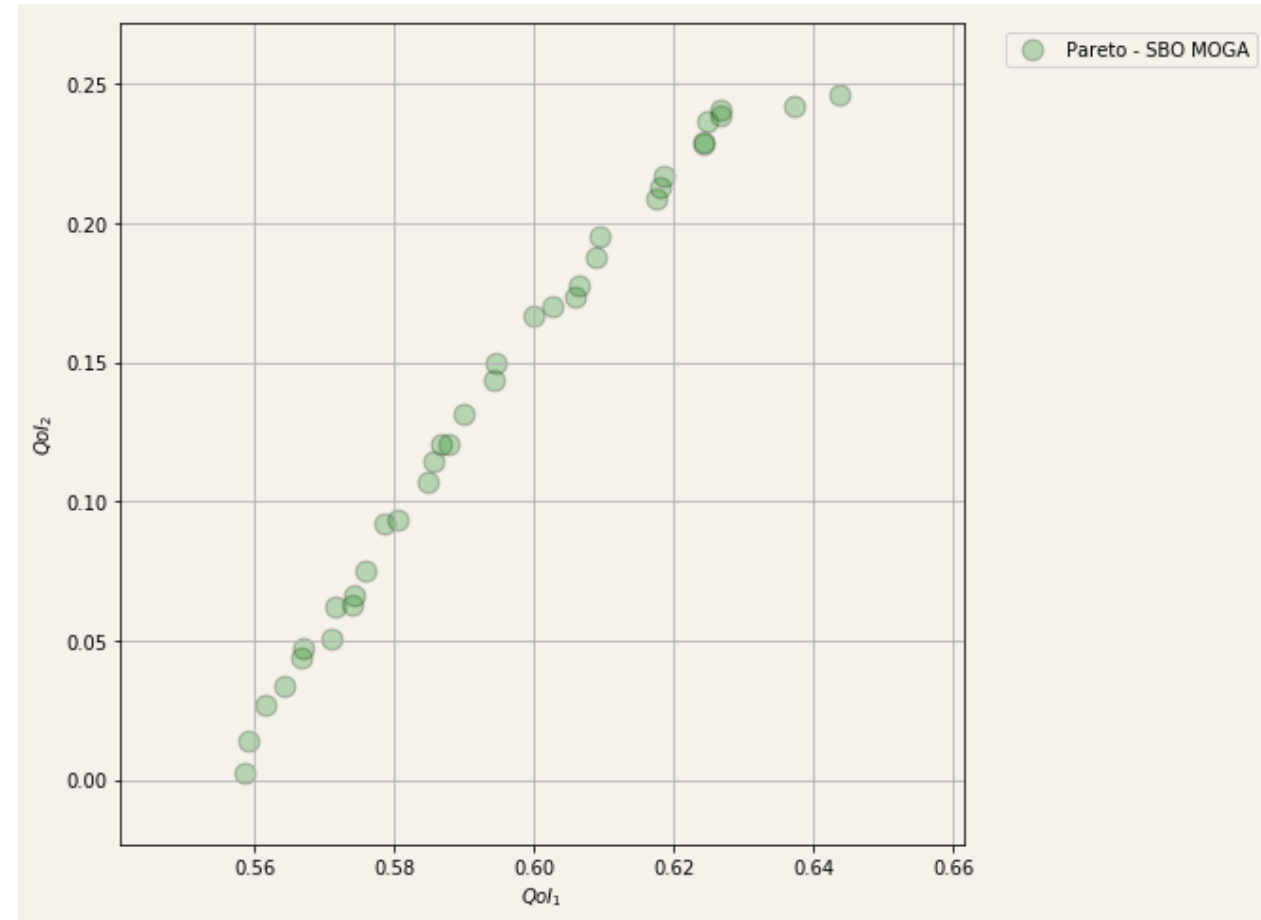
R correlation matrix for all data points. This term is constructed using a Gaussian correlation function.

f response values vector (training data)

G matrix of trend functions at all data points

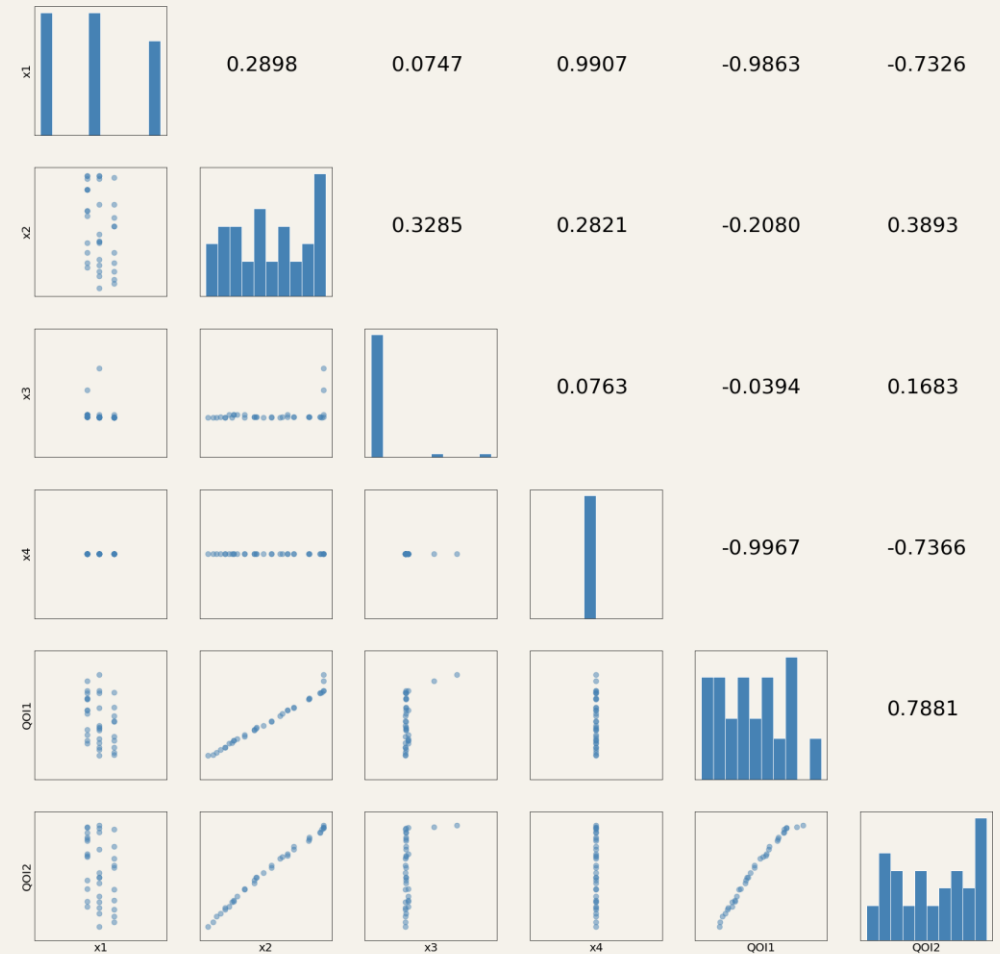
CFD optimization sample cases

- After constructing the surrogate, we can use any optimization method.
- In this case, we conducted a multi-objective optimization (MOO) study, where we aimed at minimizing the drag and maximizing the lift.
- We also used a non-linear constraint to ensure that lift is positive.
- The surrogate was constructed using 600 observations.
- During the MOO study we limited the number of maximum iterations to 4000.
- Doing 4000 iterations at the surrogate level is inexpensive.
- The final output of the MOO is the Pareto front, which represents the frontier in the design space where we can not get any more improvement.



CFD optimization sample cases

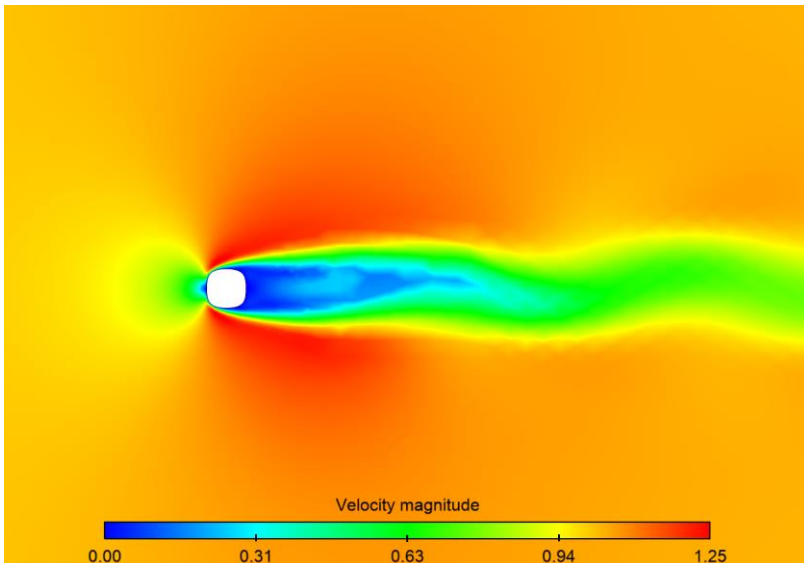
- Each point in the pareto front (objective function design space), has a corresponding value in the input variables design space.
- For each point in the pareto front we know the corresponding design variables.
- It is not recommended to construct a surrogate or conduct statistical analysis of skew or bias data.
- In this case, the output of the MOO study is already biased towards the optimal values.



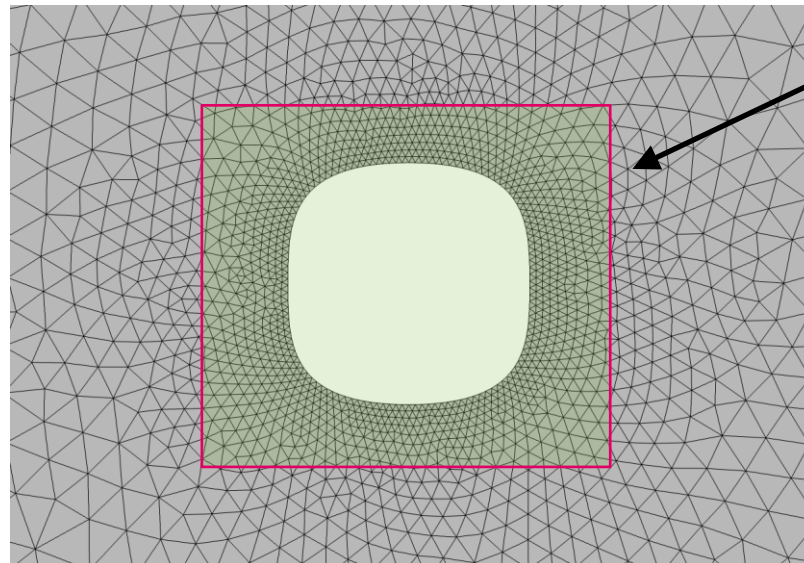
CFD optimization sample cases

- Adjoint optimization:

- In adjoint optimization, we first need to compute a CFD solution and the QOI.
- With this solution, we then compute the adjoint solution which will give us all the derivatives of the CFD solution with respect to the QOI.
- Remember, the adjoint solution depends on the state of the CFD solution.
- Then we can proceed to design a new shape using the shape sensitivity vectors.
- The shape sensitivity vectors tell us where and how to deform the mesh, not how much we need to deform it.
- To control the mesh deformation, we create a control box where we define many points that will control how to deform the mesh. We might need to add geometrical constraints to limit the deformations or avoid unrealistic deformations.

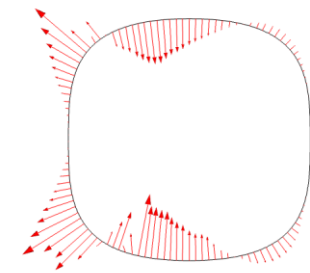


CFD solution



Mesh with control box

Mesh control box

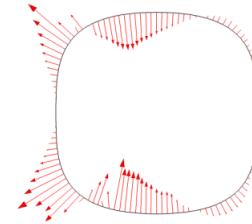
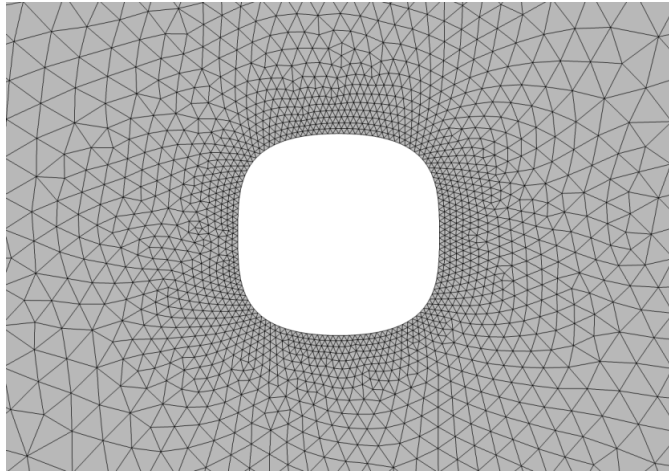
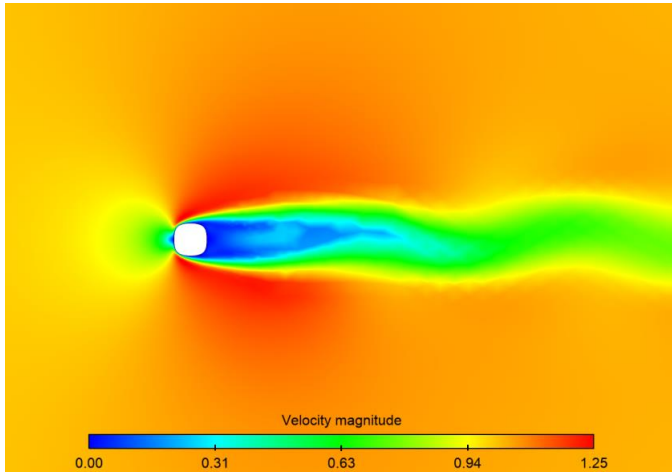


Shape sensitivity vectors

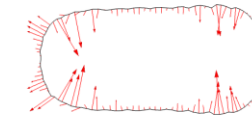
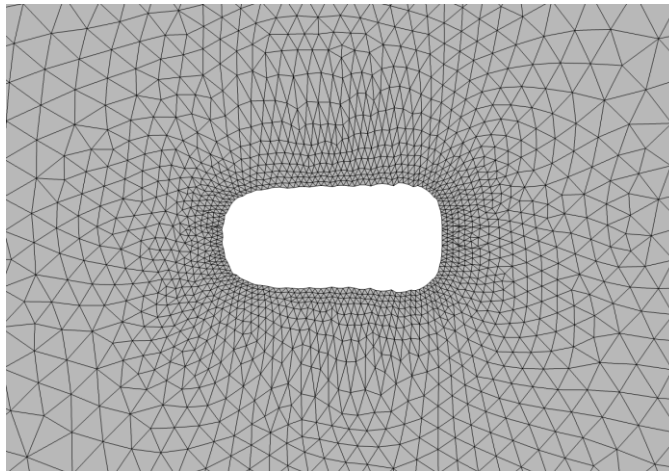
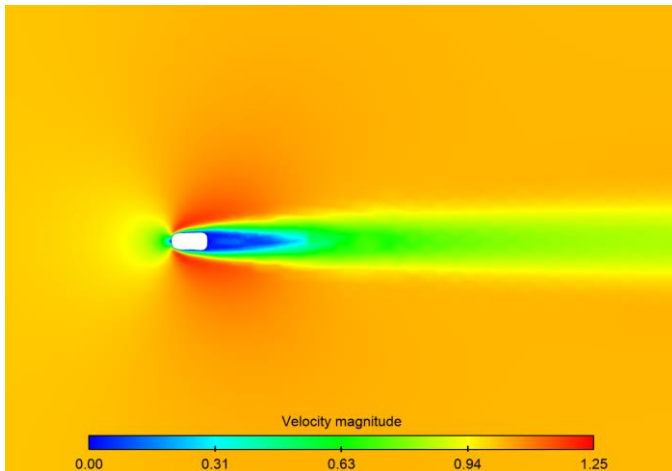
CFD optimization sample cases

- To arrive to the desired value or shape we will need to conduct many design iterations.
- During these design iterations, it might happen that the quality of the mesh will be low (negative volumes, high skewness, high non-orthogonality)
- Therefore, we will need to re-mesh the domain and map the solution.

Iteration 1

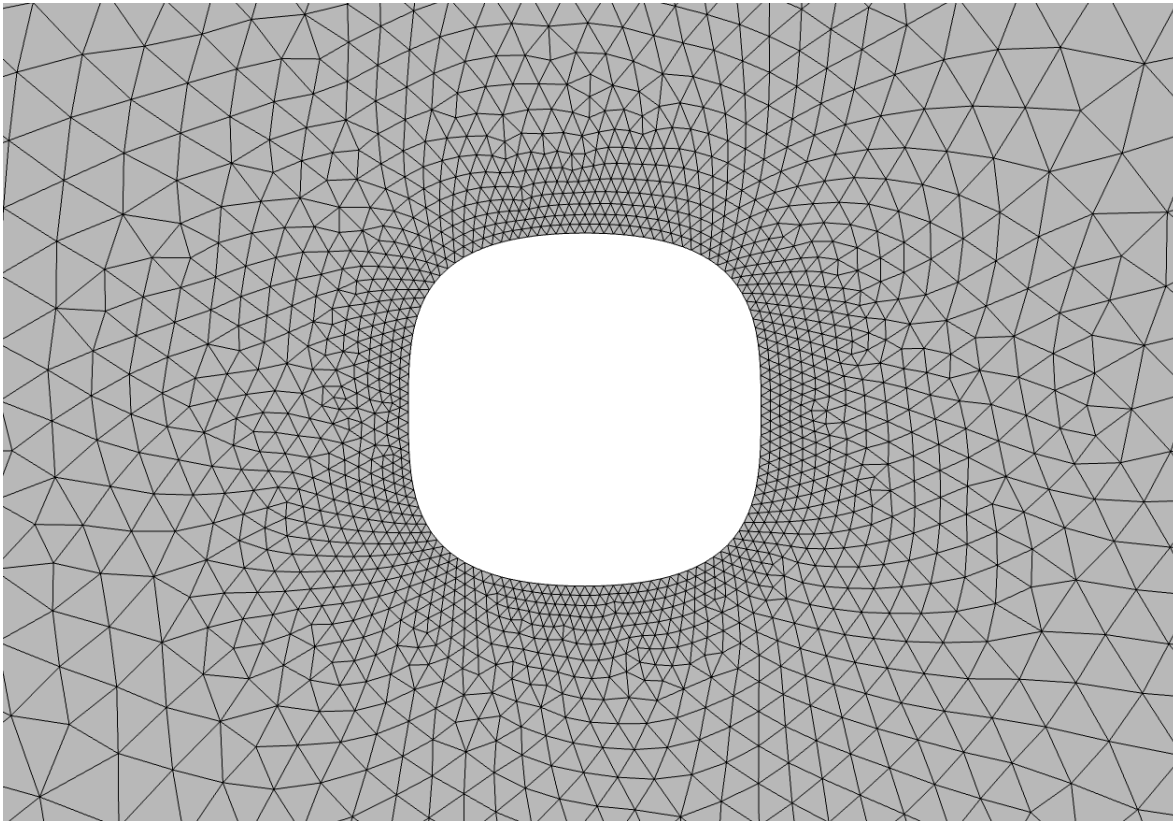


Iteration 15

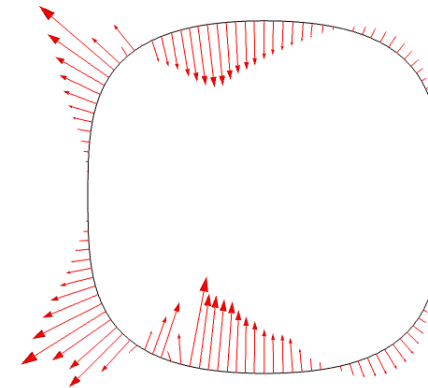


CFD optimization sample cases

- To avoid low quality meshes or a noise surface mesh (as in this case), it is recommended to limit the shape change between iterations.
- In this case, we were aiming at a 10% improvement of the QOI between iterations.
- Assuming that the behavior of the QOI is linear (which is not true at all), the mesh is deformed by displacing the surface nodes a distance that will give the desired improvement.



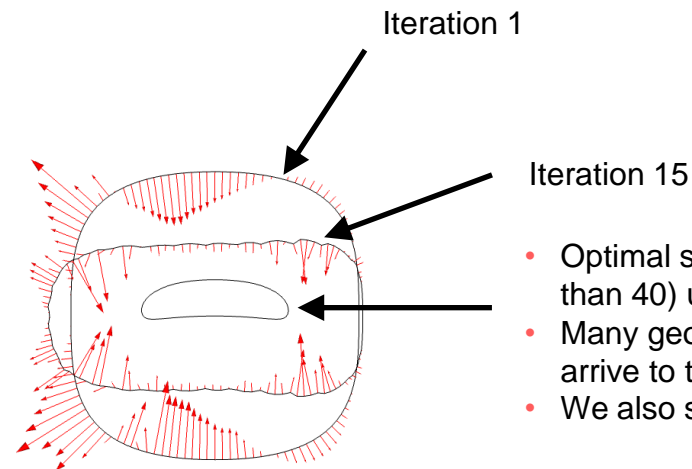
<http://www.wolfdynamics.com/training/opt/image10.gif>



<http://www.wolfdynamics.com/training/opt/image11.gif>

CFD optimization sample cases

- As already mentioned, an adjoint solver takes a flow solution and calculate the sensitivity of the QOI in response to all the inputs of the system, simultaneously, in a single computation.
- These sensitivities gives designers key information on how to modify the shape of the body.
- Sometimes, the insight gained by adjoint solvers can take the design in unexpected directions or can take longer to converge in comparison to parameter-based optimization methods.
- In this case, and after 15 iterations we are still far from the optimal solution.
- Adjoint optimization is very well fitted for fine tuning.
- In this case, it might have been wiser to use the adjoint method to fine tune the optimal solution obtained using parameter-based optimization methods.



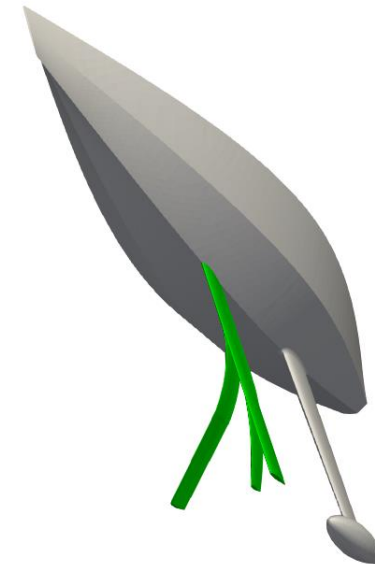
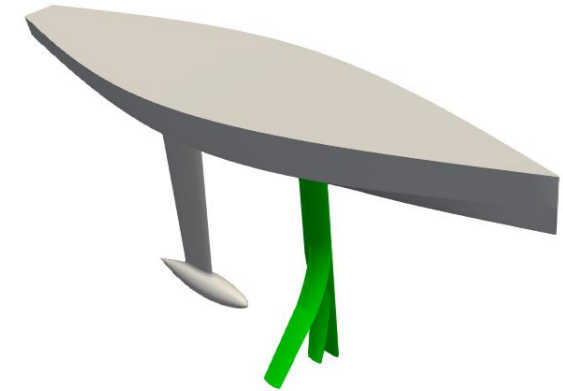
- Optimal solution obtained after many iterations (more than 40) using a gradient-based solver.
- Many geometrical constraints were used in order to arrive to this solution.
- We also smoothed the geometry and the mesh.

CFD optimization sample cases:
Daggerboard shape optimization

CFD optimization sample cases

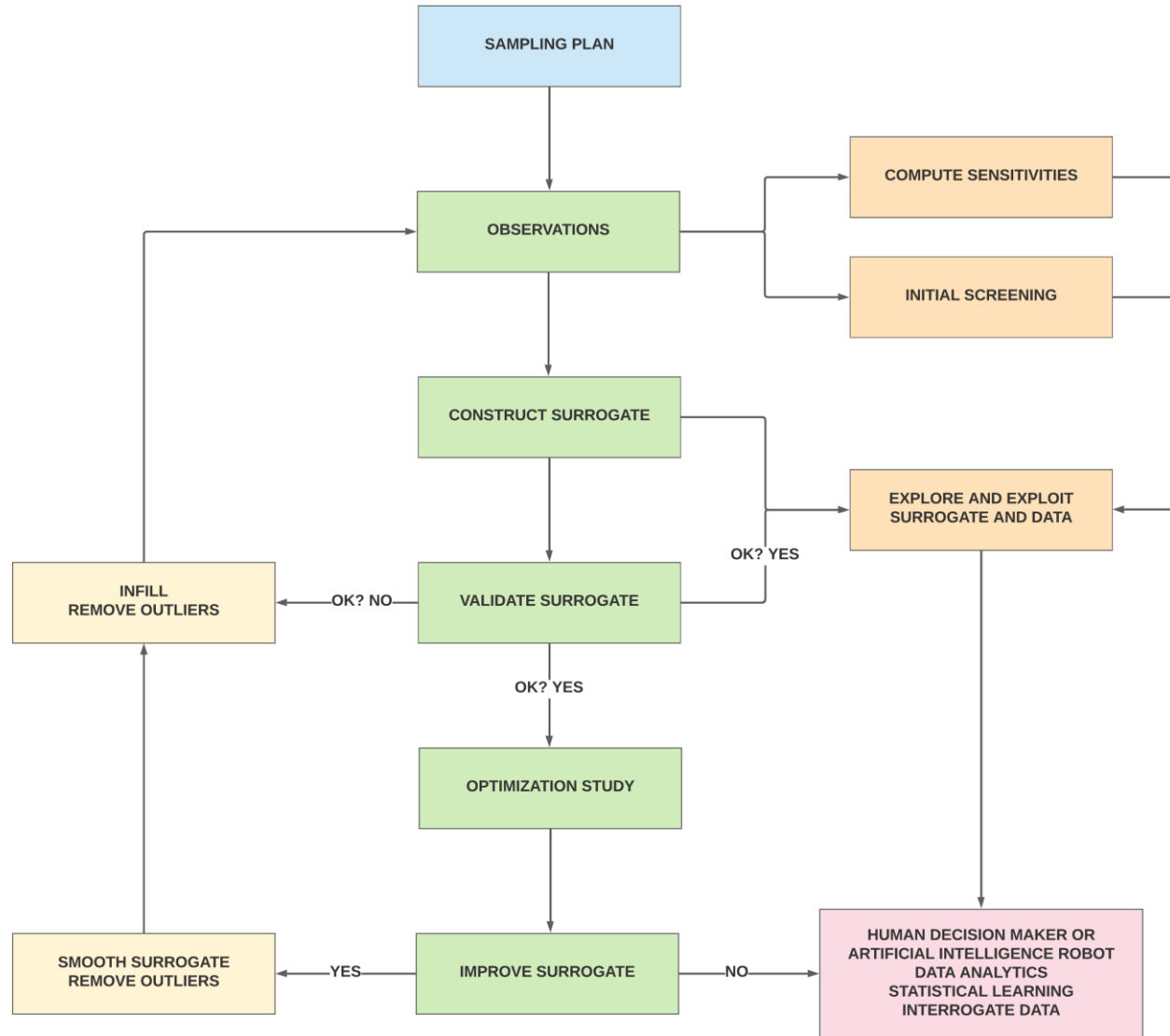
- Daggerboard shape optimization:

- In this case we aim at optimizing the shape of a daggerboard.
- The goals are maximizing the vertical force and minimizing the drag coefficient.
- There are 12 design variables and 1 non-linear constraint (the lateral force on the daggerboard).
- All design variables are bounded and for the non-linear constraint we use an inequality.
- The daggerboard is divided in three sections.
- The design variables control the airfoil shape (NACA 6-Series and NACA 4-Series) and the daggerboard taper, sweep and dihedral (flexion).
- To conduct the MOO we used the MOGA (multi-objective genetic algorithm) method and SBO.



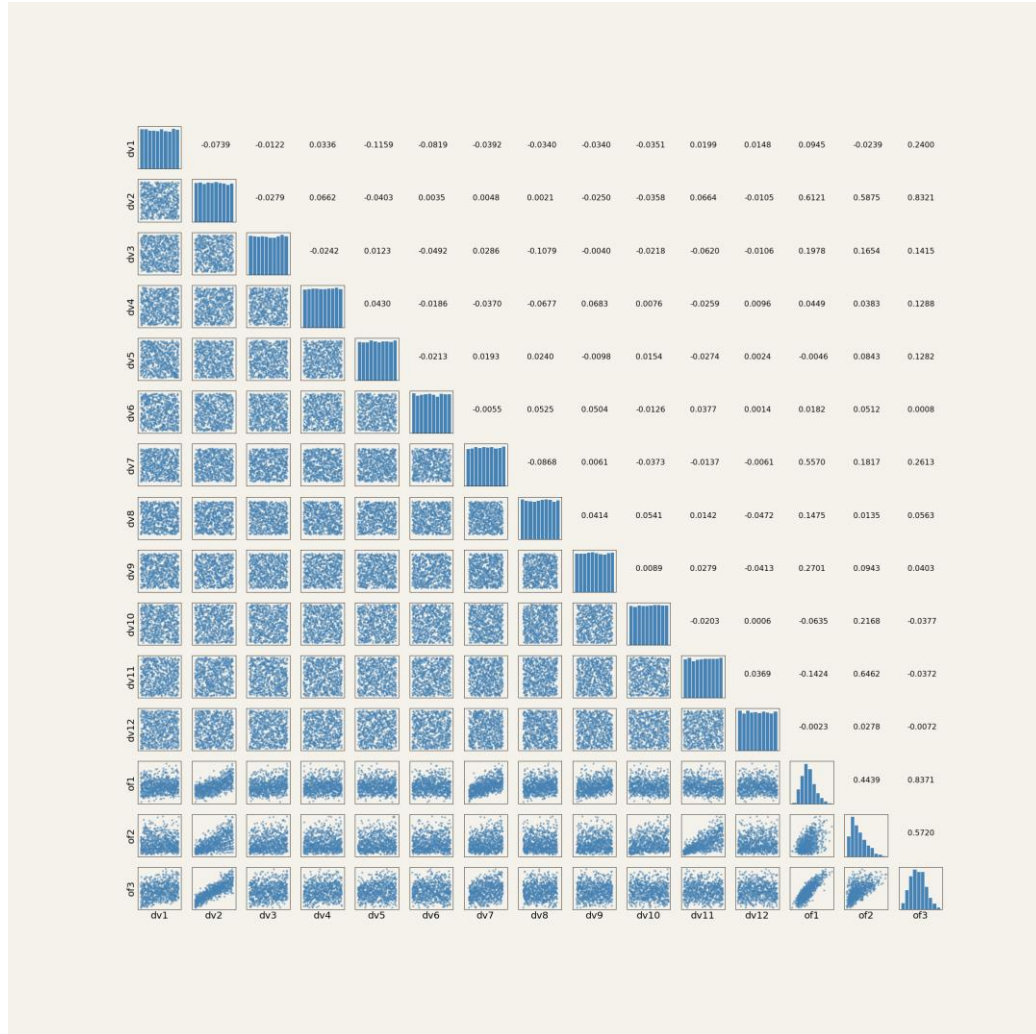
CFD optimization sample cases

- Surrogate-based optimization (SBO) workflow.

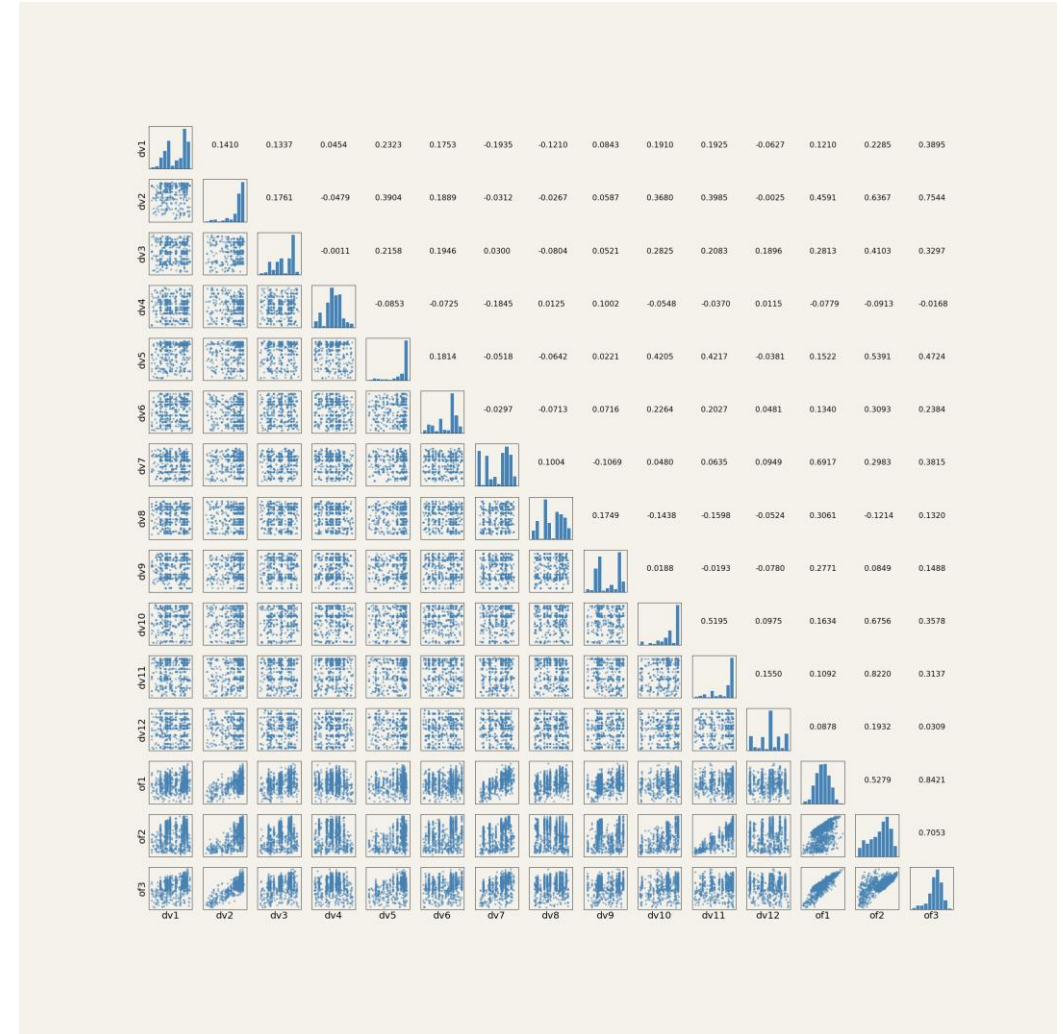


CFD optimization sample cases

- Scatter matrix plot of the DSE study and the MOO study using the MOGA method (EA).



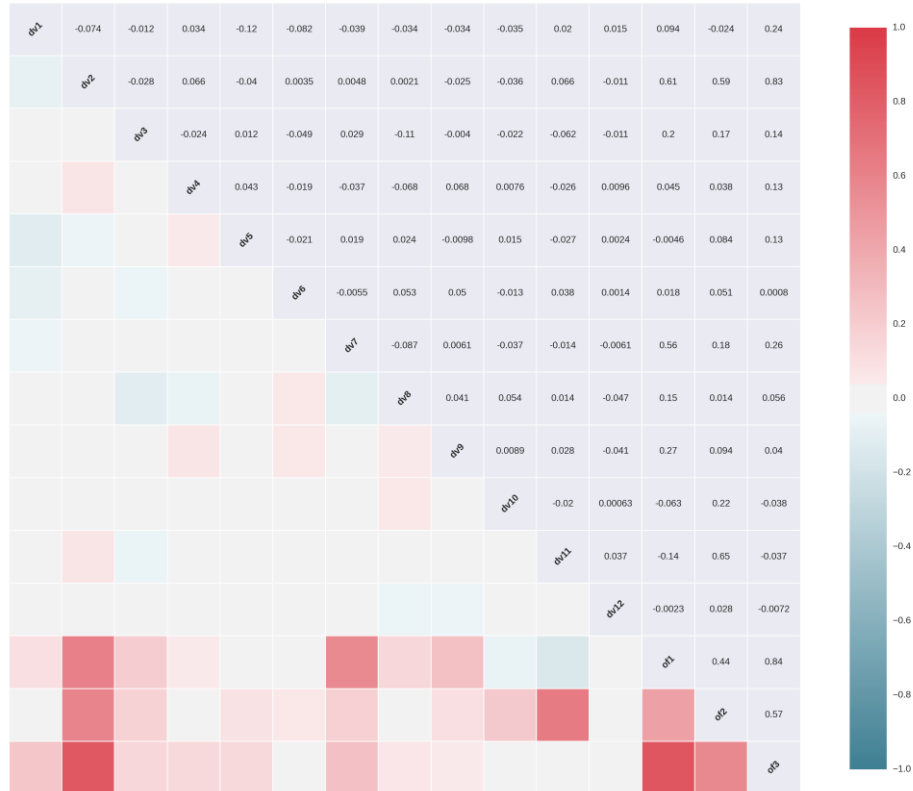
Scatter matrix plot – DACE 750 experiments



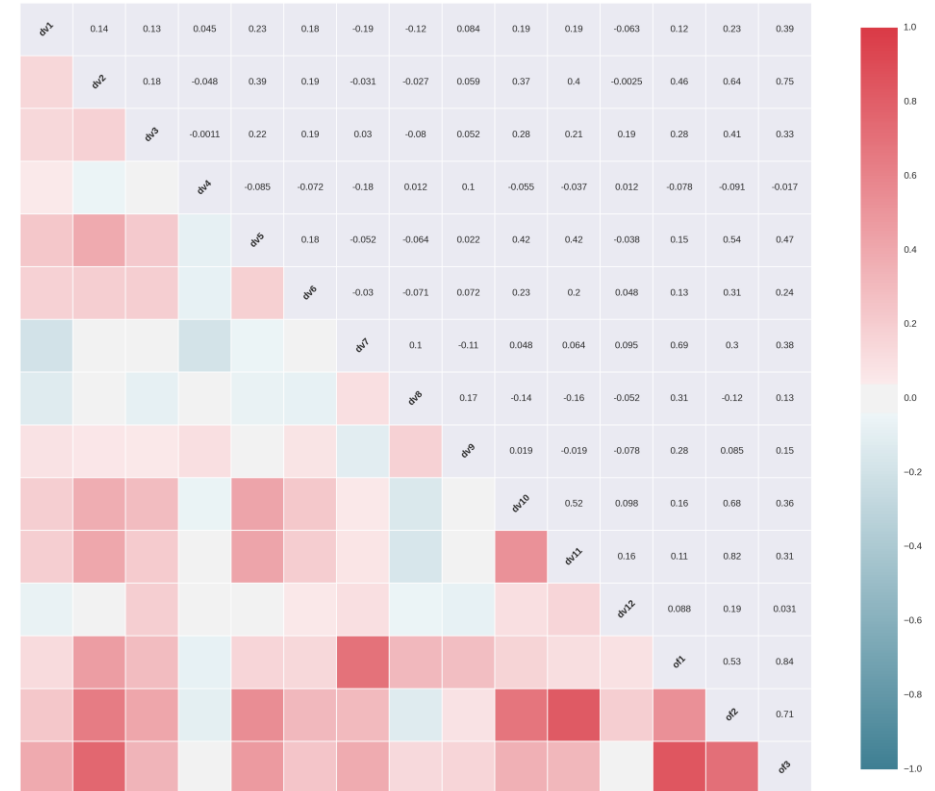
Scatter matrix plot – MOGA (EA) – 1500 experiments

CFD optimization sample cases

- Correlation matrices of the DSE study and the MOO study using the MOGA method (EA).

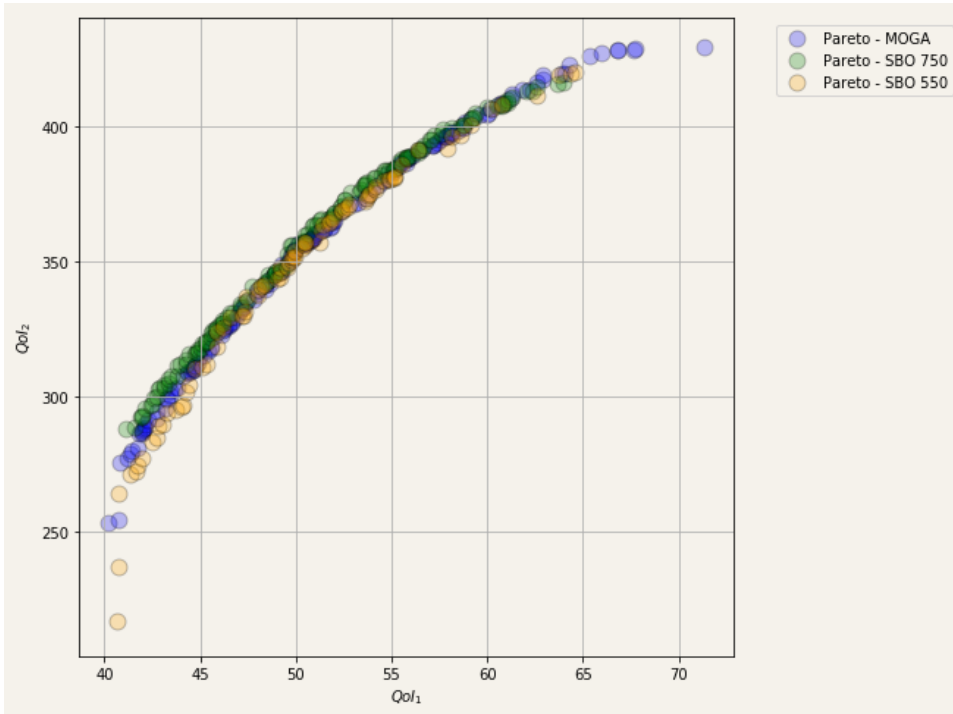


Correlation matrix plot – DACE 770 experiments

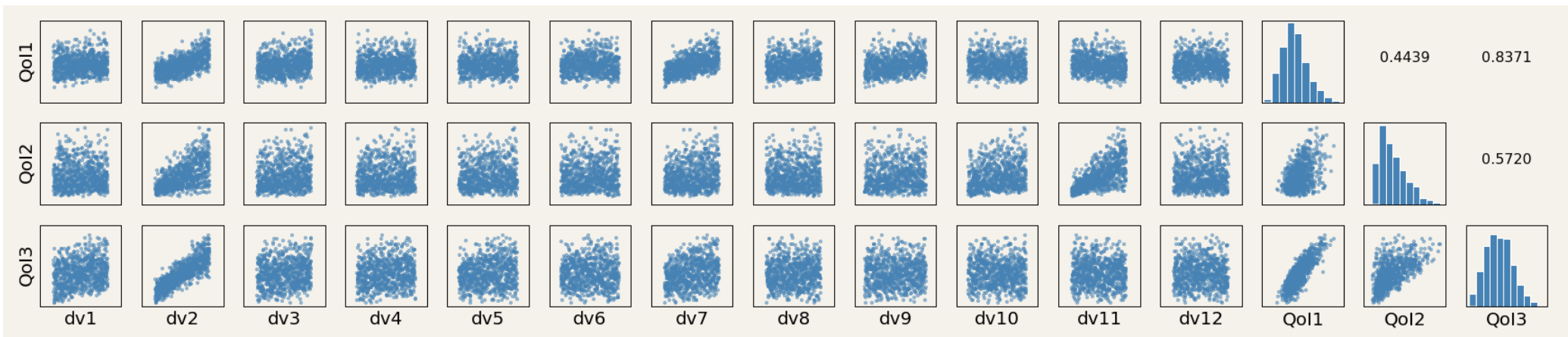


Correlation matrix plot – MOGA (EA)

CFD optimization sample cases

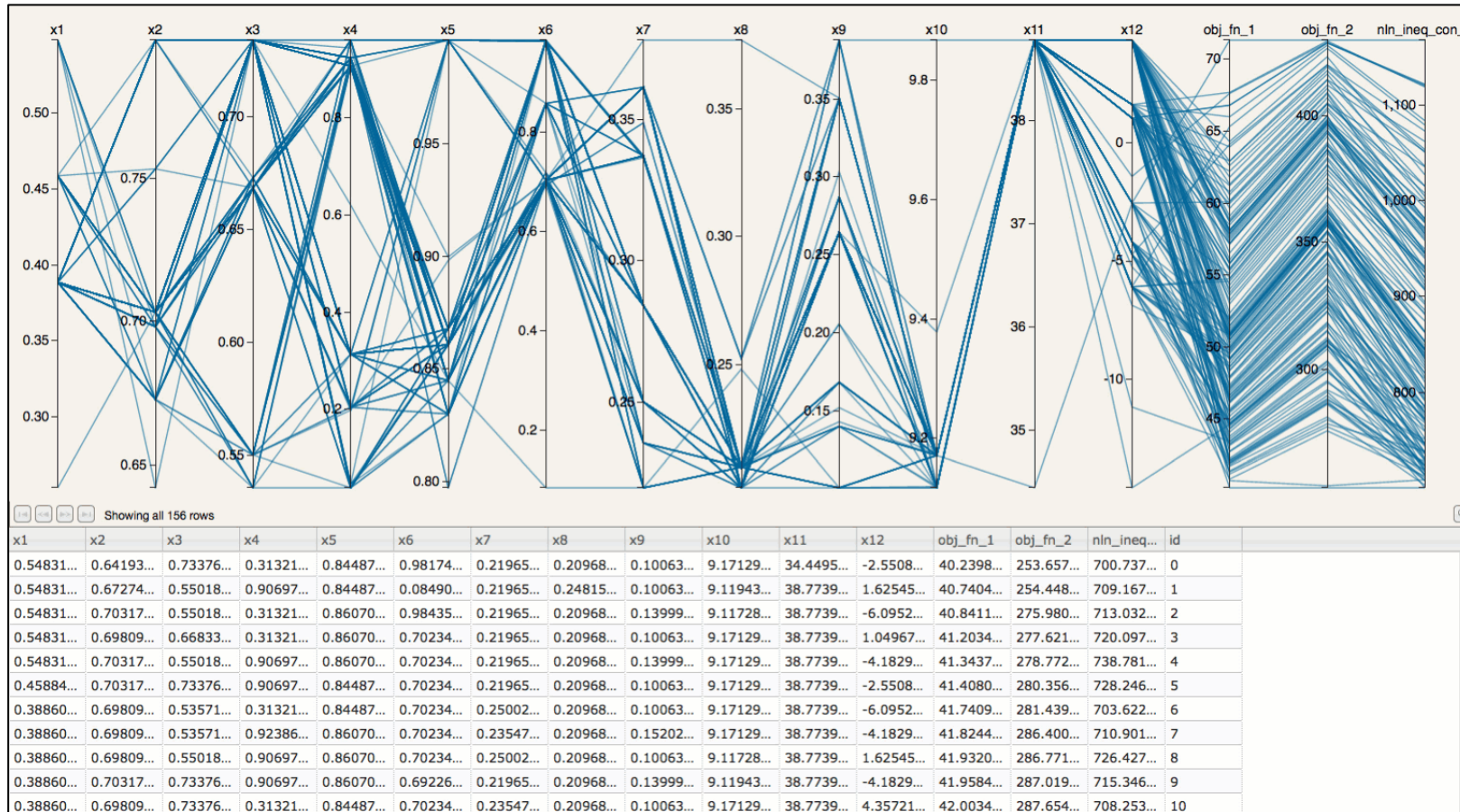


- Pareto front and overall response of the design space.
- The MOGA pareto was constructed using 1500 high-fidelity simulations and a genetic algorithm.
- The SBO 750 pareto was constructed using 750 high-fidelity simulations and kriging interpolation.
- The SBO 550 pareto was constructed using 550 high fidelity simulations and kriging interpolation.
- By using SBO we were able to obtain similar response in half the computational time.
- Plus the insight gained from the DSE study.



CFD optimization sample cases

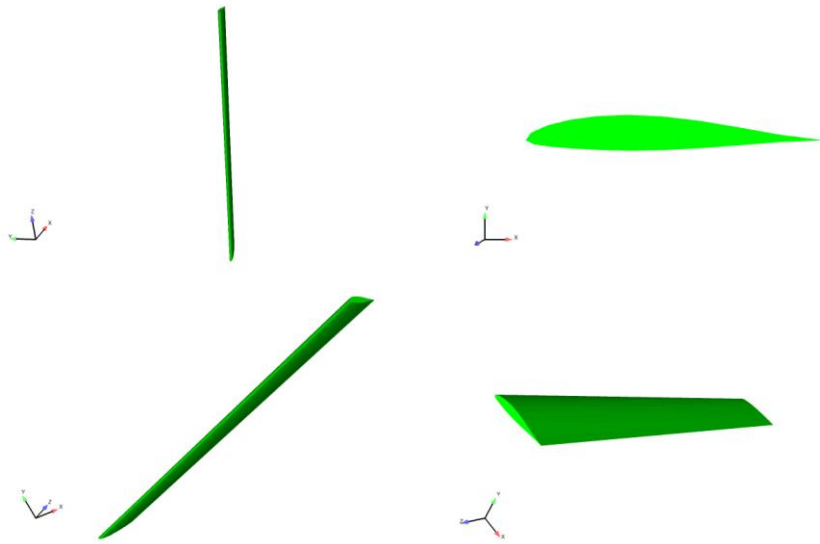
- When working with multi-dimensional data, the best way to explore the data is by using interactive parallel coordinates.
- These plots let us easily find correlations between design variables and system responses.



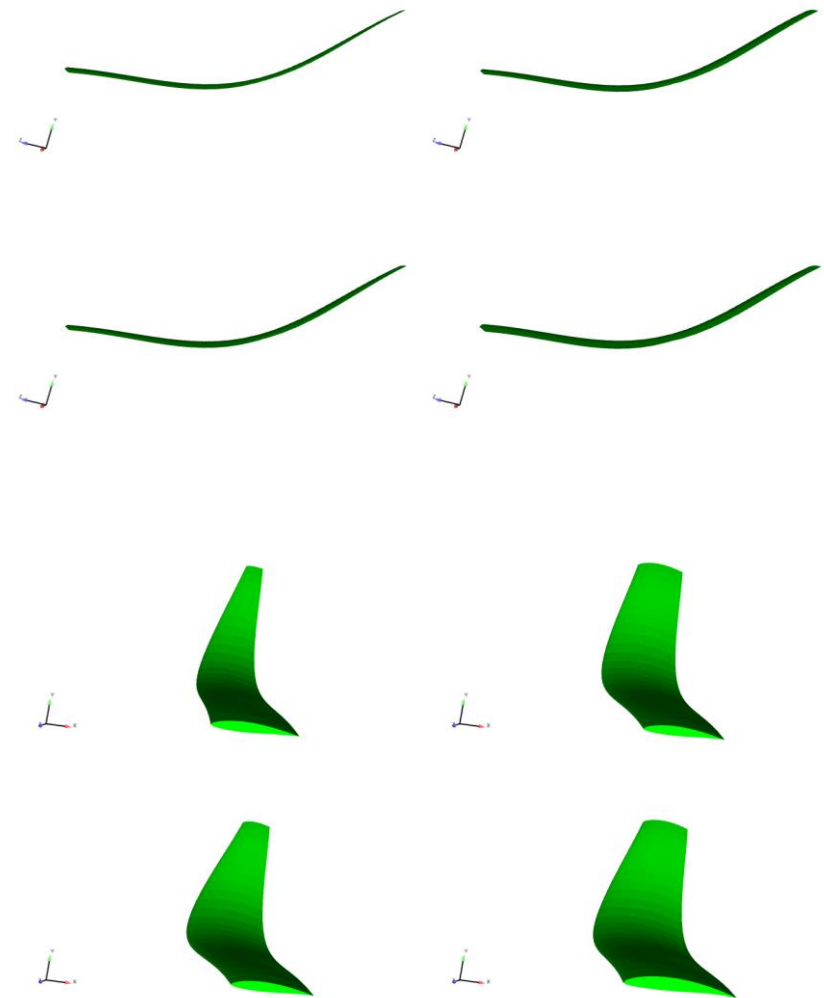
Interactive parallel coordinates

<http://www.wolfdynamics.com/training/opt/image3.gif>

CFD optimization sample cases



Initial geometry

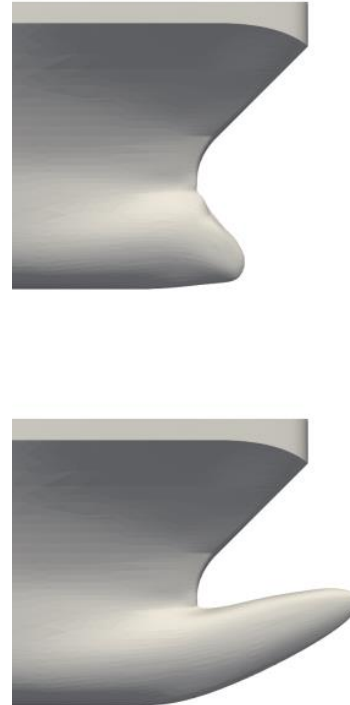
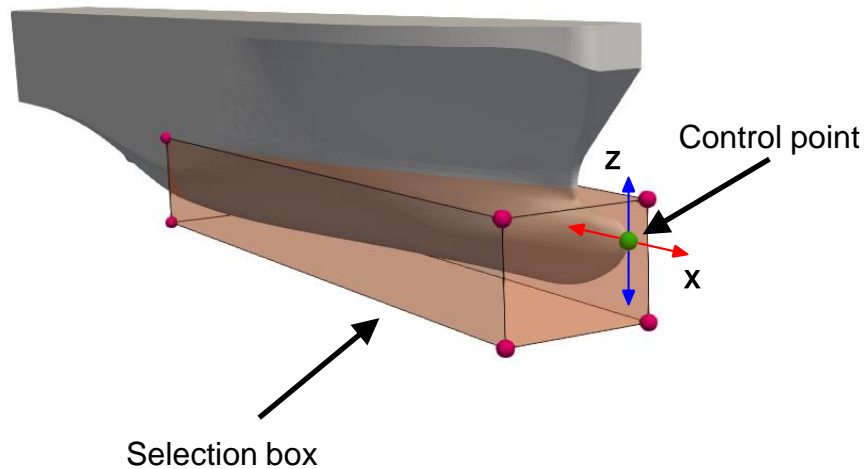


Optimal candidates

Four non-dominated solutions belonging to the pareto

CFD optimization sample cases:
Bulbous bow shape optimization

CFD optimization sample cases



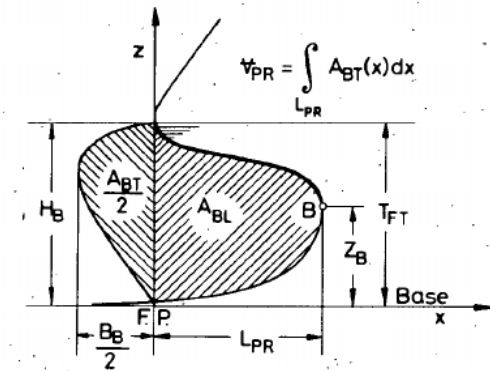
- Shape optimization of a bulbous bow.
- The geometry was parametrized following the guidelines given by Kracht*.
- Hereafter we only consider two parameters; protrusion and immersion.

$$C_{LPR} = \frac{L_{PR}}{L_{PP}}$$

Protrusion

$$C_{ZB} = \frac{Z_B}{T_{FT}}$$

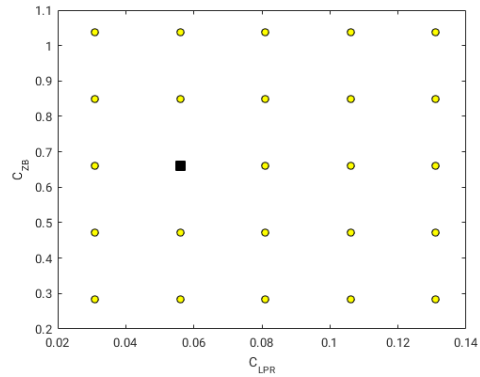
Immersion



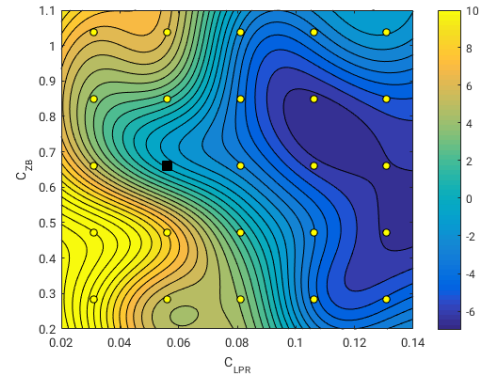
- One single control point is enough to deform the bulbous bow.
- The goal is to minimize the drag.

* A. M. Kracht, Design of bulbous bow, SNAME Transactions 86 (1978) 197–217.

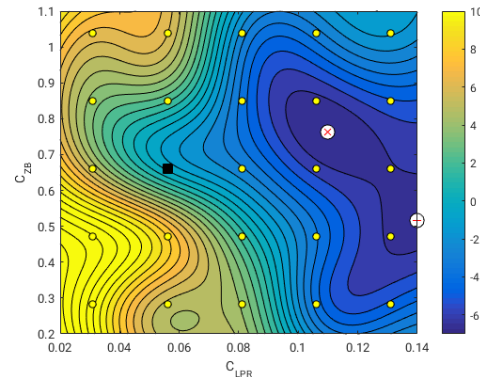
CFD optimization sample cases



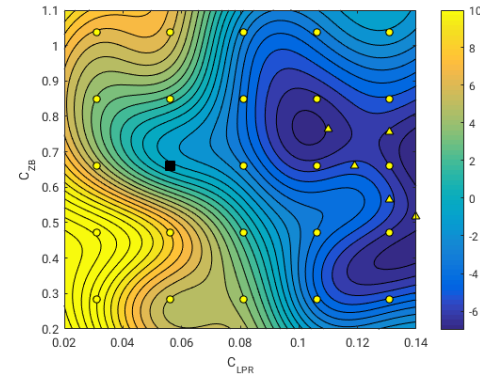
Sampling plan



Construct surrogate



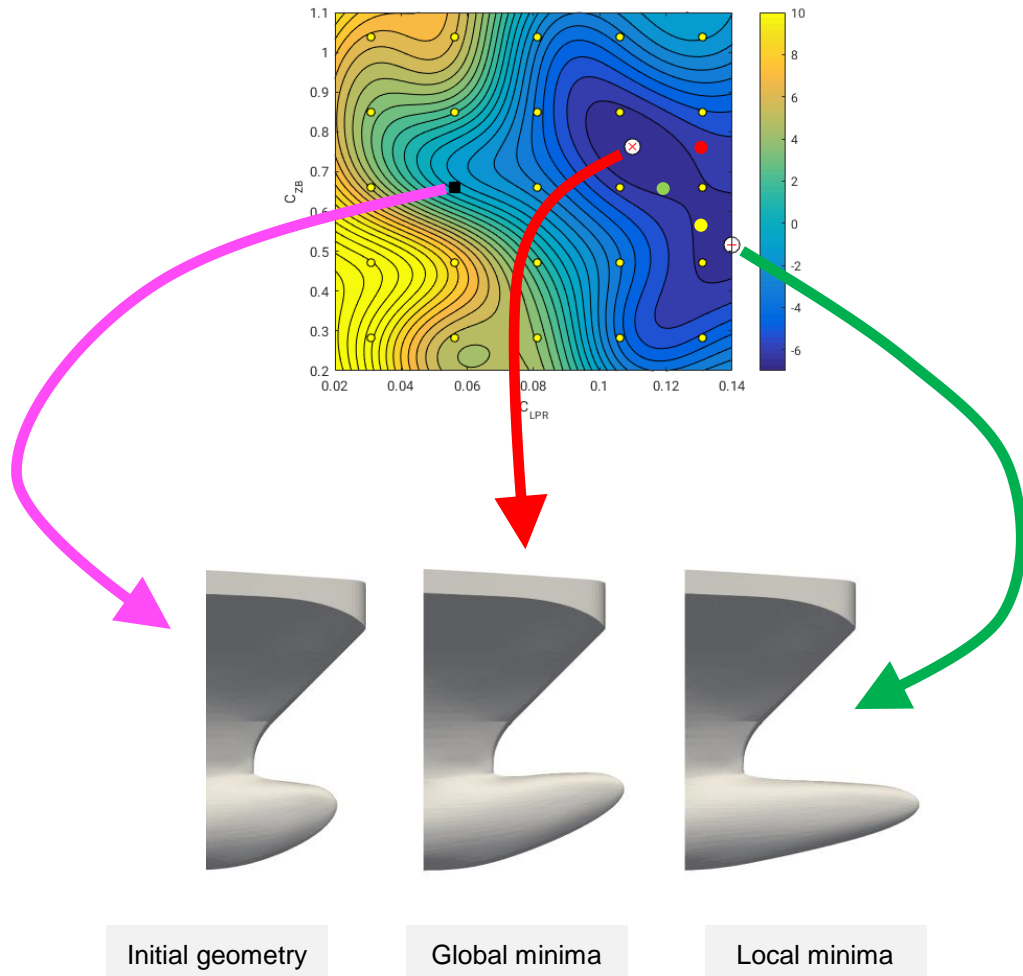
Exploit and optimize



Improve (infill)

- Shape optimization study conducted using surrogate-based optimization (SBO)
- Full-factorial experiment with 25 observations.
- Surrogate constructed using kriging interpolation (no need to smooth the surrogate).
- Surrogate optimized using gradient-based and derivative-free methods.
- We found one global minima and one local minima in a valley region (multi-modal problem).
- We used as infill points the minimum and three points located in the valley.
- The cost of constructing the surrogate is much less than optimizing at the high-fidelity level.

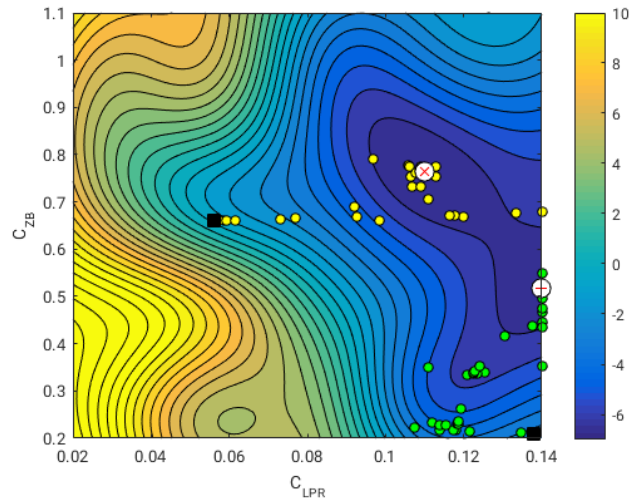
CFD optimization sample cases



C_{LPR}	C_{ZB}	Resistance reduction	Percentage error	Note
0.119	0.660	$\cong 5\%$	$\cong 7\%$	●
0.131	0.566	$\cong 5\%$	$\cong 6\%$	●
0.131	0.755	$\cong 5\%$	$\cong 6\%$	●
0.109947	0.762845	$\cong 7\%$	$\cong 2\%$	Global minima
0.14	0.515651	$\cong 6\%$	$\cong 2\%$	Local minima

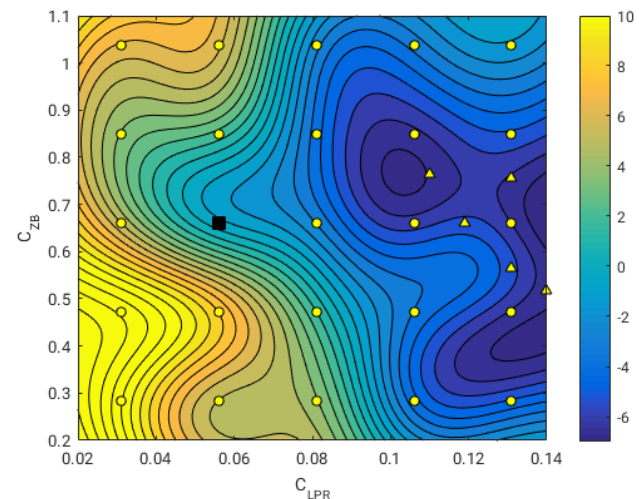
- The percentage error is computed in reference to the predicted value at the surrogate level and the outcome of the high-fidelity simulations.
- All the simulations were run in an out-of-the-box workstation with 16 procs and 128 GB RAM.
- Asynchronous simulations – Many simulations running concurrently using 4 procs.
- Average time per simulation: 3 to 5 hours

CFD optimization sample cases



Optimization method iterations

- Gradient-based method with multi-start (MFD).
- Approximately 40 iterations for each optimization path (function and gradient evaluations).
- If we had done the simulations at the high-fidelity level, the optimization loop total time would have been much larger.
- No possibility of running concurrent simulations (we are limited by the maximum number of gradients that can be computed).
- A lot of guessing involved (starting point, step-size, and so on)

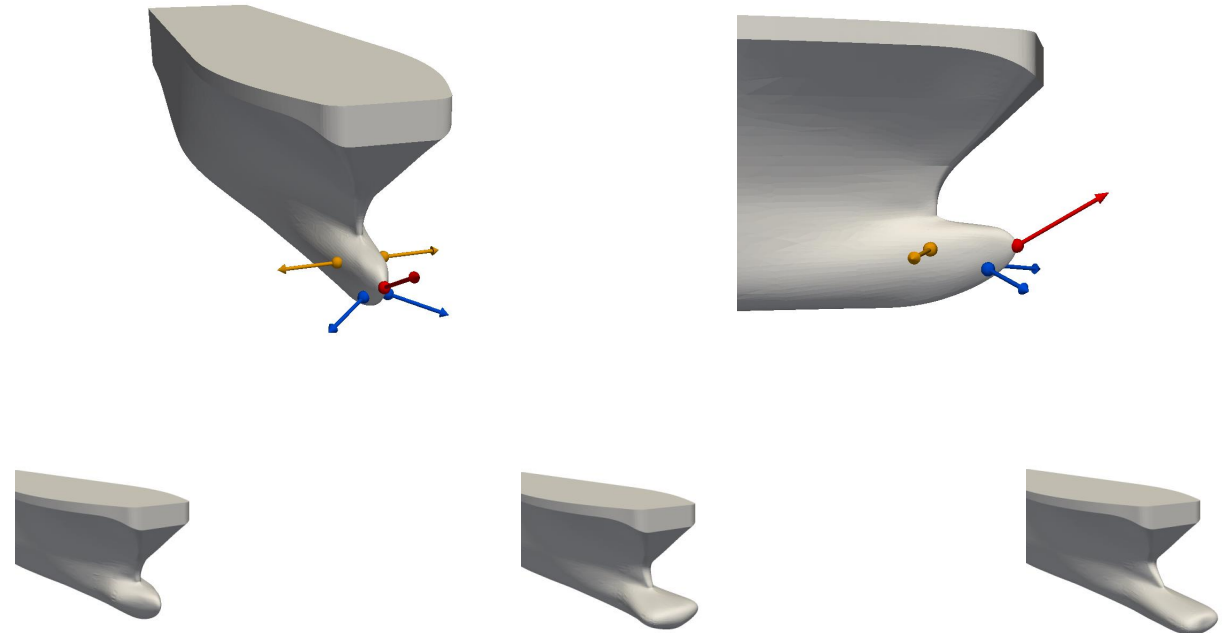
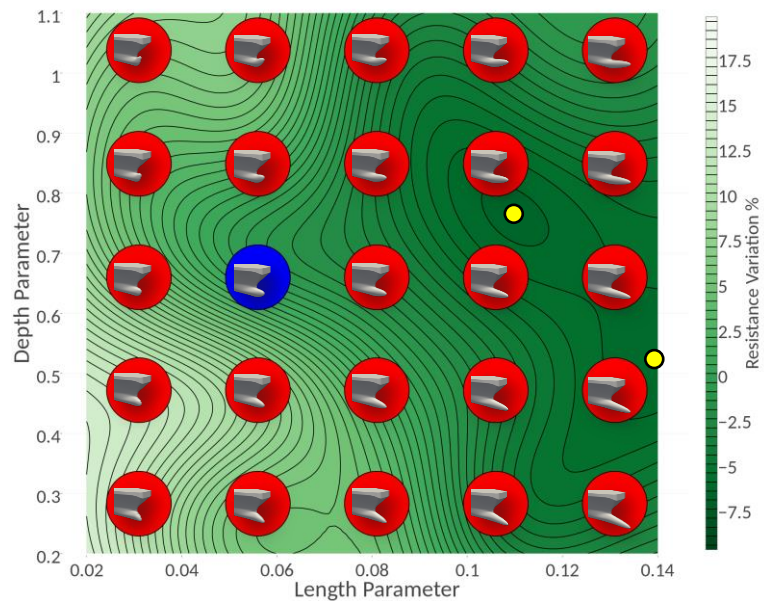


Surrogate infilling

- Infilling of the design space using five new training points.
- Two training points correspond to the minimum values.
- Three training points are located in the valley region.
- The new global minima still is located in the same region, with a percentage error of less than 1%.
- The region where the local minima is located is different, we might need to add new training points or smooth the surrogate.
- In any case we focus our attention in the global minima region.

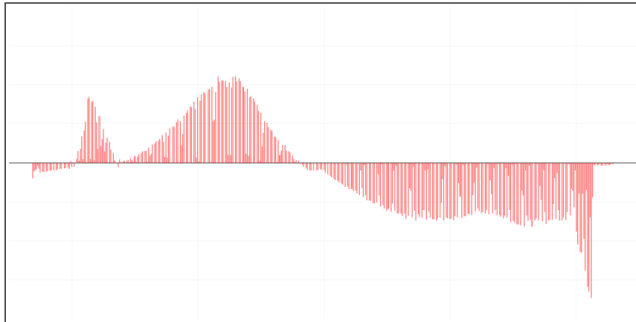
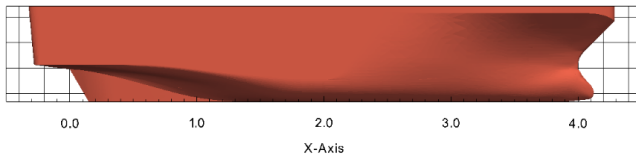
CFD optimization sample cases

- Engineering design optimization can very abstract (balance between aesthetics and usability, operational requirements, manufacturing process, and so on). So, not necessarily the minimum will be the final solution.
- By exploring and visualizing the surrogate and according to the subjective criteria of the designer, alternatives candidates can be identified.
- We can also cross-relate the information gathered to explore different design spaces.

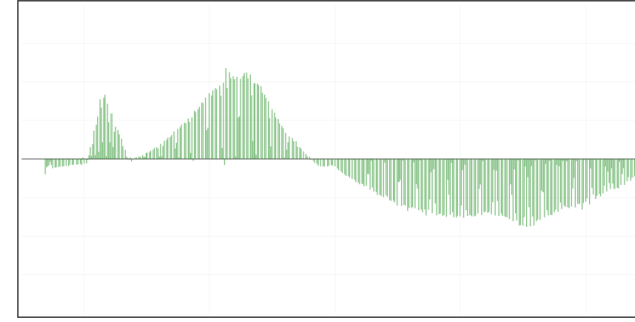
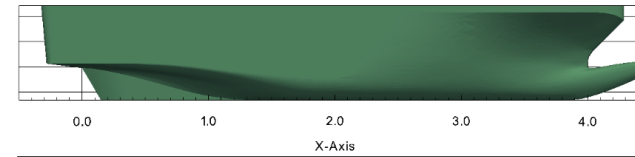


CFD optimization sample cases

KPI
 $R = 1.15$



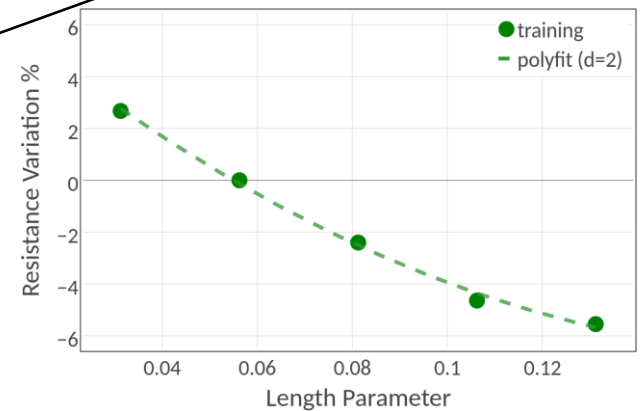
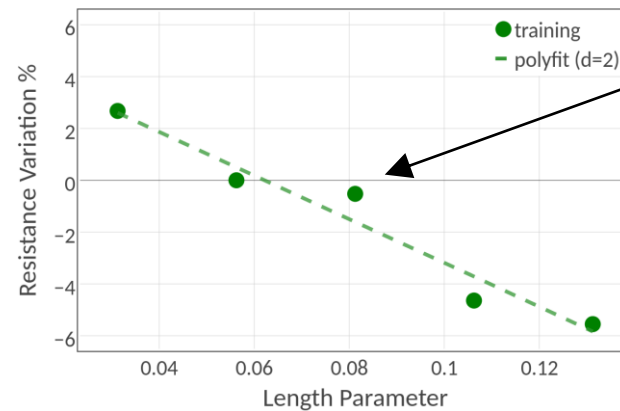
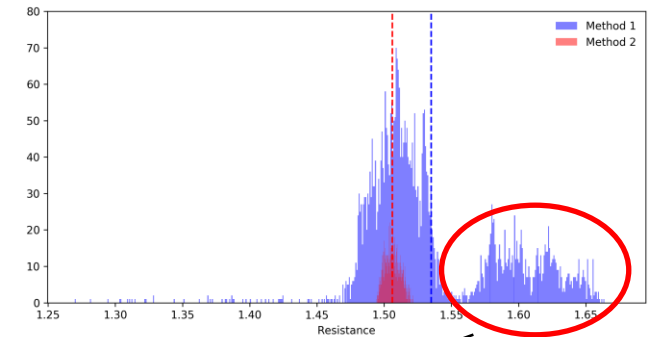
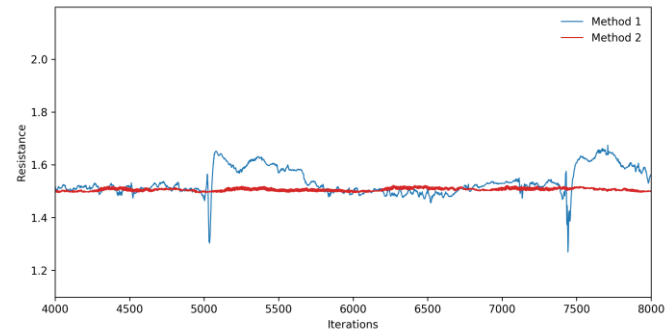
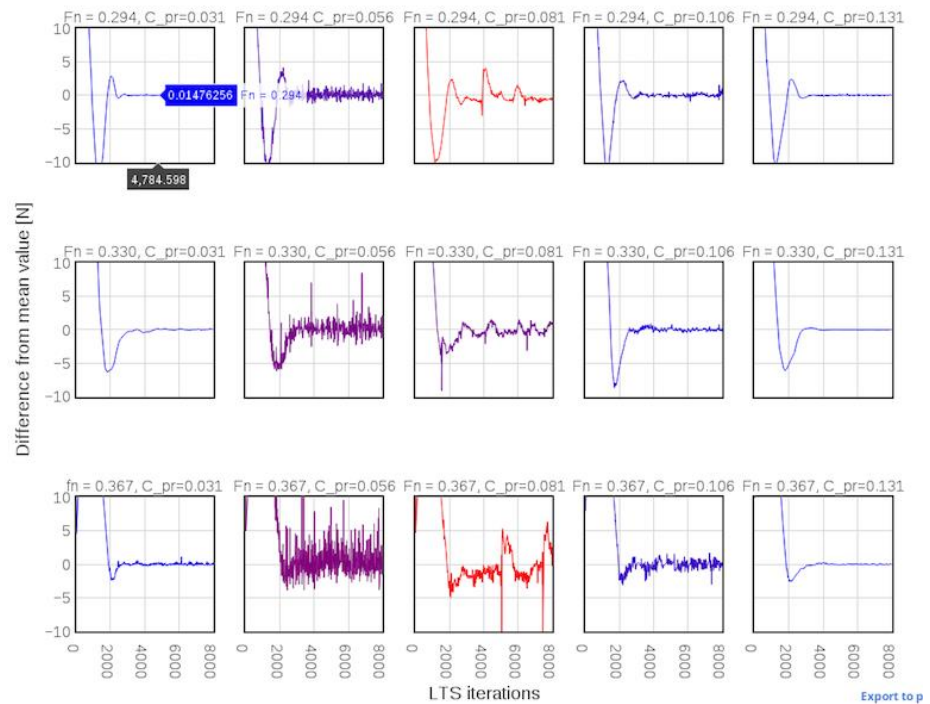
KPI
 $R = 0.92$



- Key performance indicators (KPI), are not necessarily your final solution.
- Remember to look data from different angles (aggregate, transform, and cluster data).

CFD optimization sample cases

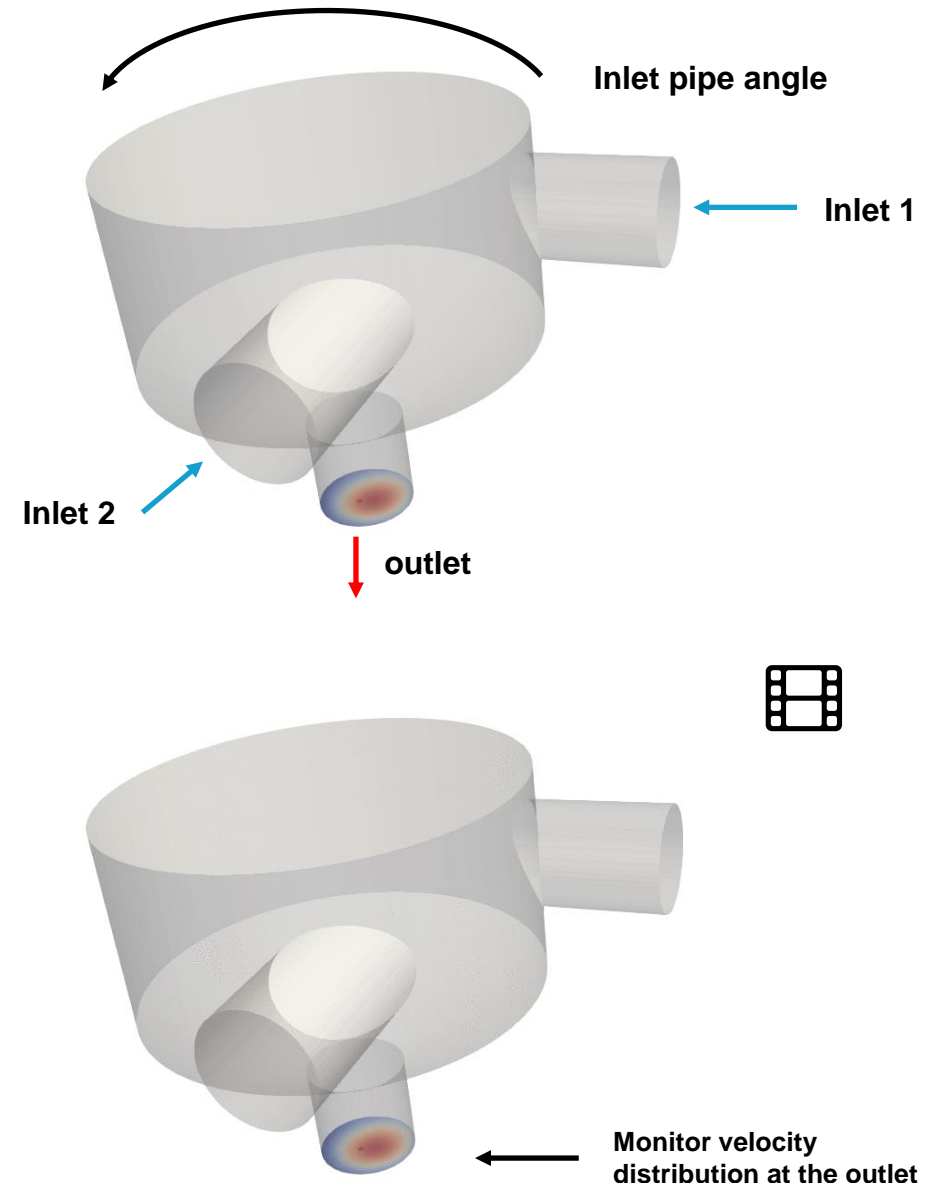
- Data aggregation, machine learning, and real-time data analytics can be used to identify trends, patterns and outliers.
- Corrections can be done on-the-fly.



CFD optimization sample cases:
Static mixer optimization using image recognition

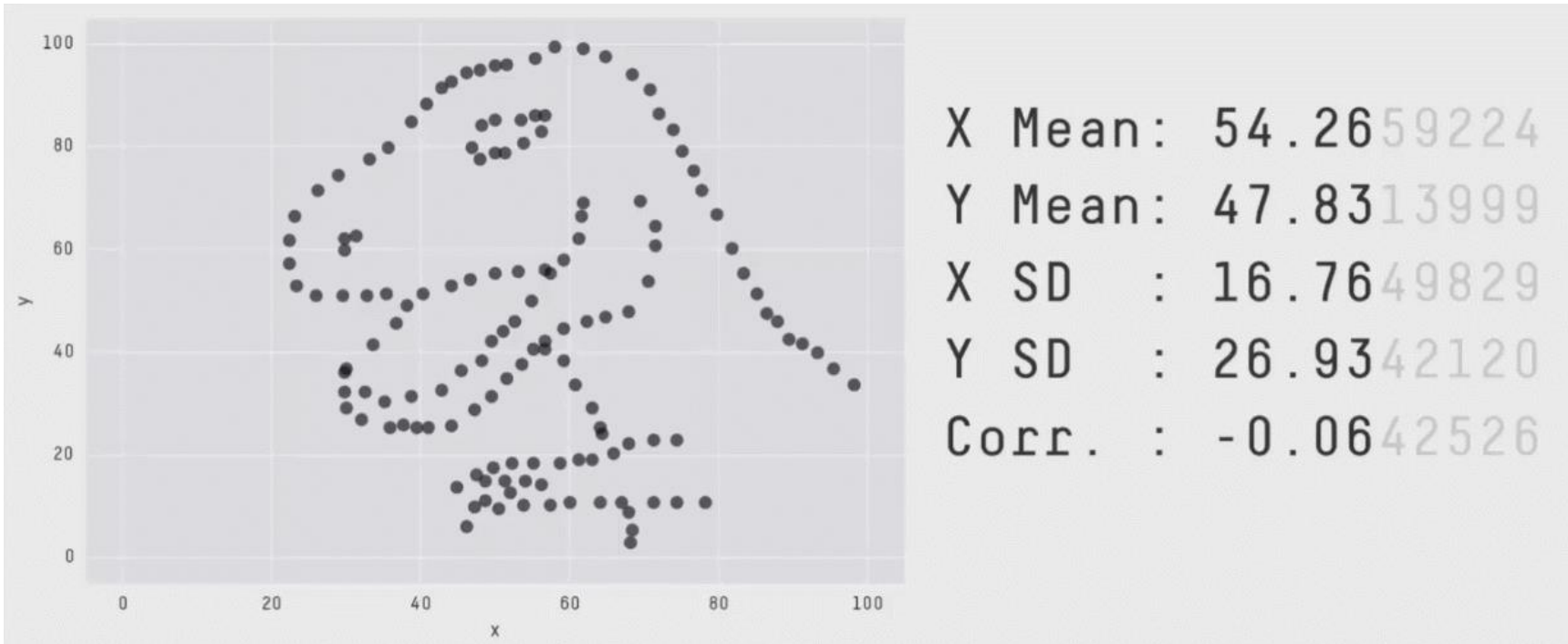
CFD optimization sample cases

- Static mixer optimization case using image recognition.
 - The main goal in this case is to obtain a given velocity distribution at the outlet by changing the angle of the inlet pipe 1.
 - The velocity distribution field at the outlet was designed in such a way that the velocity normal to the outlet surface has a paraboloid distribution.
 - Then, by using the SSIM index method we can compare the target image with current image.
- The advantage of using image recognition is that we can now fit or optimize the problem according to a given visual field (which can come from an experiment).
- This kind of problems are often optimized using integral quantities such a uniformity index, distortion coefficient, or swirl index.
- These key performance indicators (KPI) not necessarily indicate that we are satisfying a given distribution of a field variable in a given surface or section of interest.



CFD optimization sample cases

- Let us digress from the main topic to stress the importance of visualization.
- In reference [1] you can find an enlightening discussion about the importance of visualizing the data.
- In the datasaurus dataset used in reference [1], we can see how the data points morph from one shape to another, all the while maintaining the same summary statistical values to two decimal places throughout the entire process.



“...make both calculations and graphs. Both sorts of output should be studied; each will contribute to understanding.”

F. J. Anscombe [2].

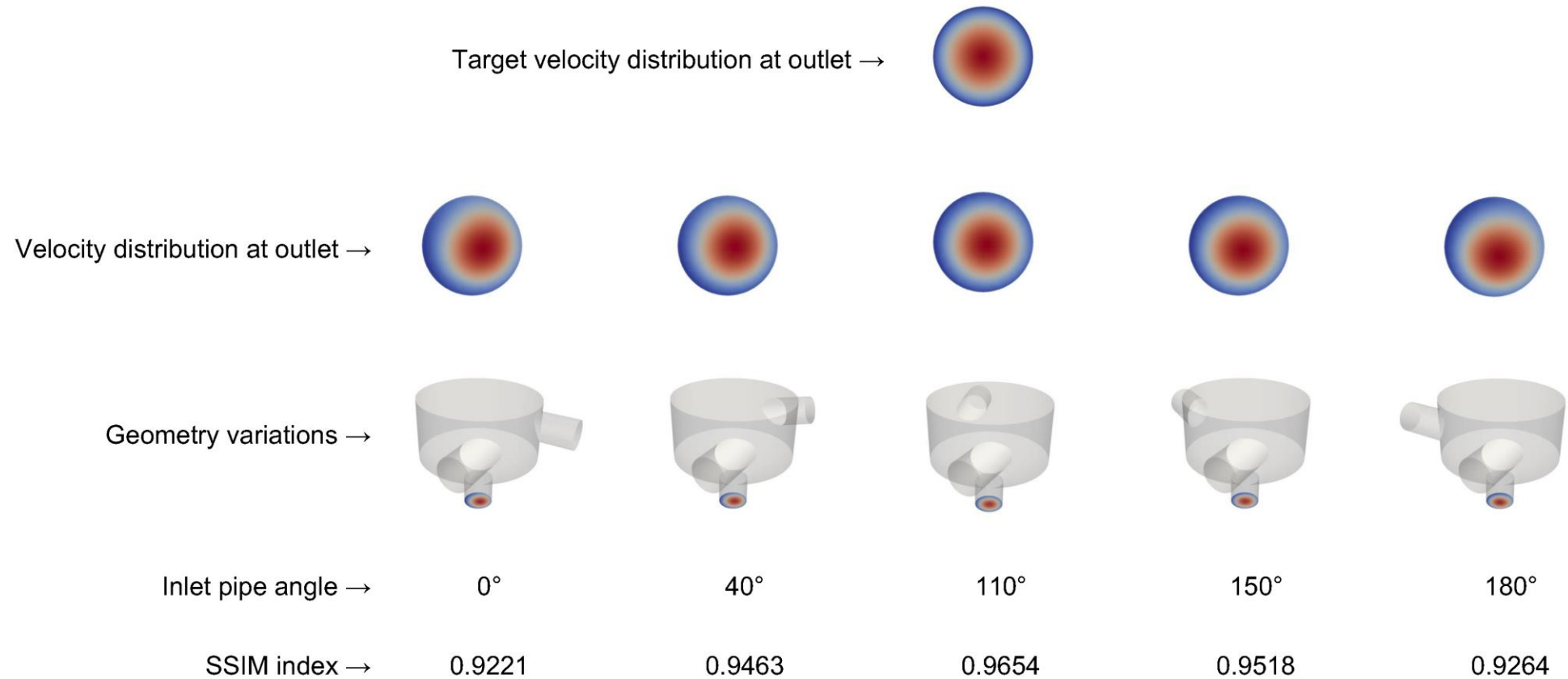
<http://www.wolfdynamics.com/training/opt/image12.gif>

[1] J. Matejka, G. Fitzmaurice. Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing. Autodesk Research. ACM SIGCHI Conference on Human Factors in Computing Systems, 2017.

[2] F. Anscombe. Graphs in Statistical Analysis. The American Statistician 27, 1, 17–21, 1973.

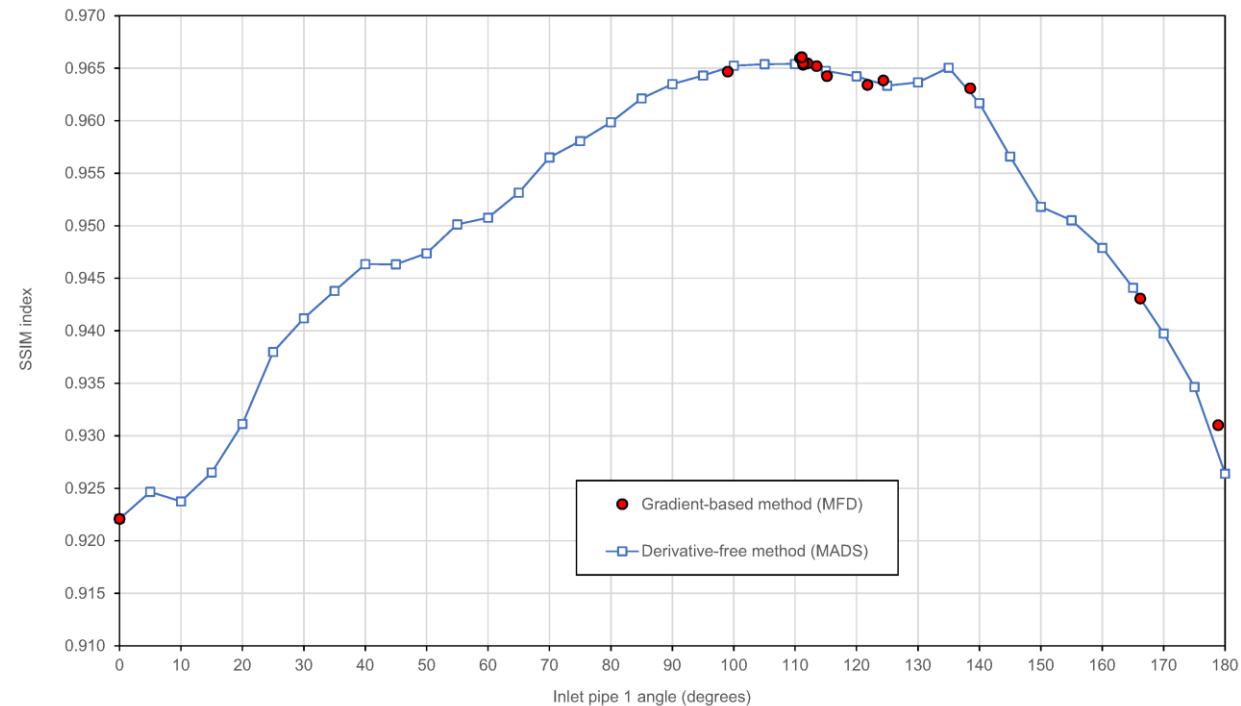
CFD optimization sample cases

- Qualitative comparison of velocity distribution at the outlet.
- The SSIM method was used to compare the images.
- The SSIM index value is bounded between 0 and 1 → A value of 1 means that the images are identical.



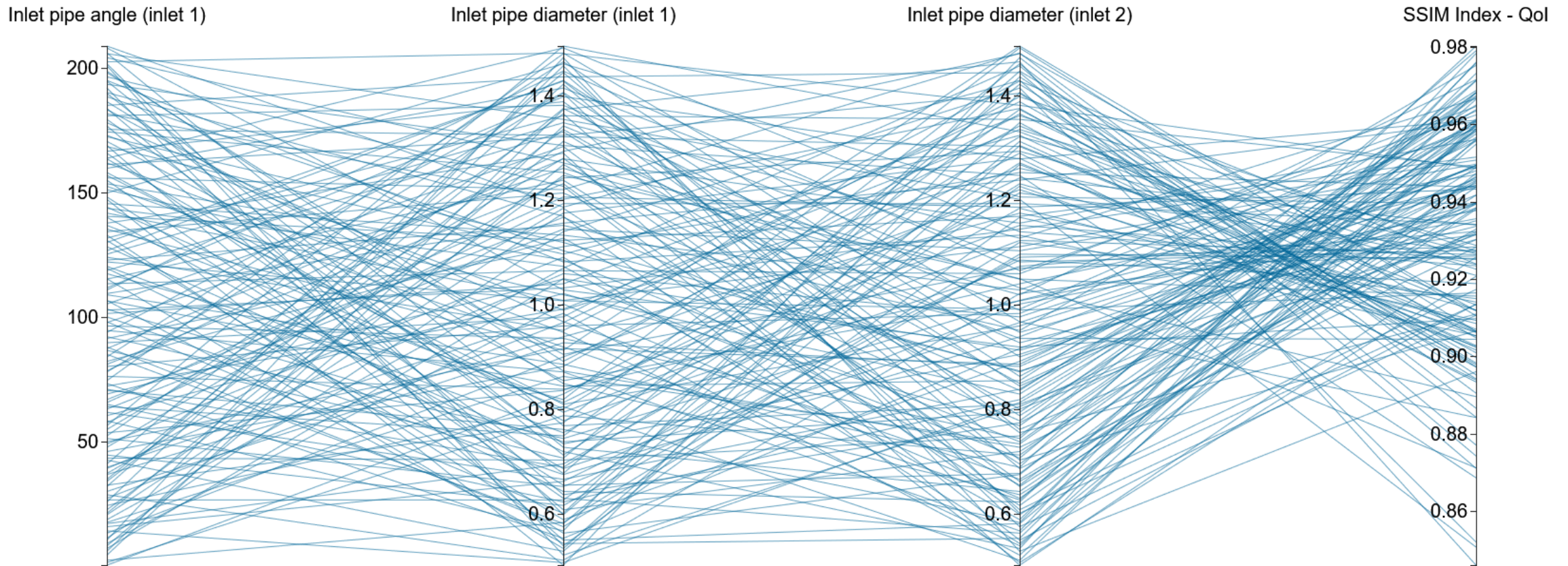
CFD optimization sample cases

- The DO study was conducted using the method of feasible directions (gradient-based method) with numerical gradients computed using forward differences.
- For the DO case, the starting point was 0 degrees, and the case converged to the optimal value in 31 function evaluations.
 - Optimal value: pipe angle equal to 111.0549 degrees and SSIM index equal to 0.9660
- In the DSE case, we explored the design space from 0 to 180 degrees, in steps of 5 degrees (36 function evaluations).
- So roughly speaking, we used the same number of function evaluations as for the DO case.
- The DSE study, while not formerly converging to the optimal solution, gives more information about the design space than the DO method.



CFD optimization sample cases

- This case can be easily extended to more design variables.
- The use of exploratory data analysis techniques is of extremely importance when studying high dimensional design spaces.
- In the figure below, the outcome of a case with three design variables is visualized using parallel coordinates (interactive).



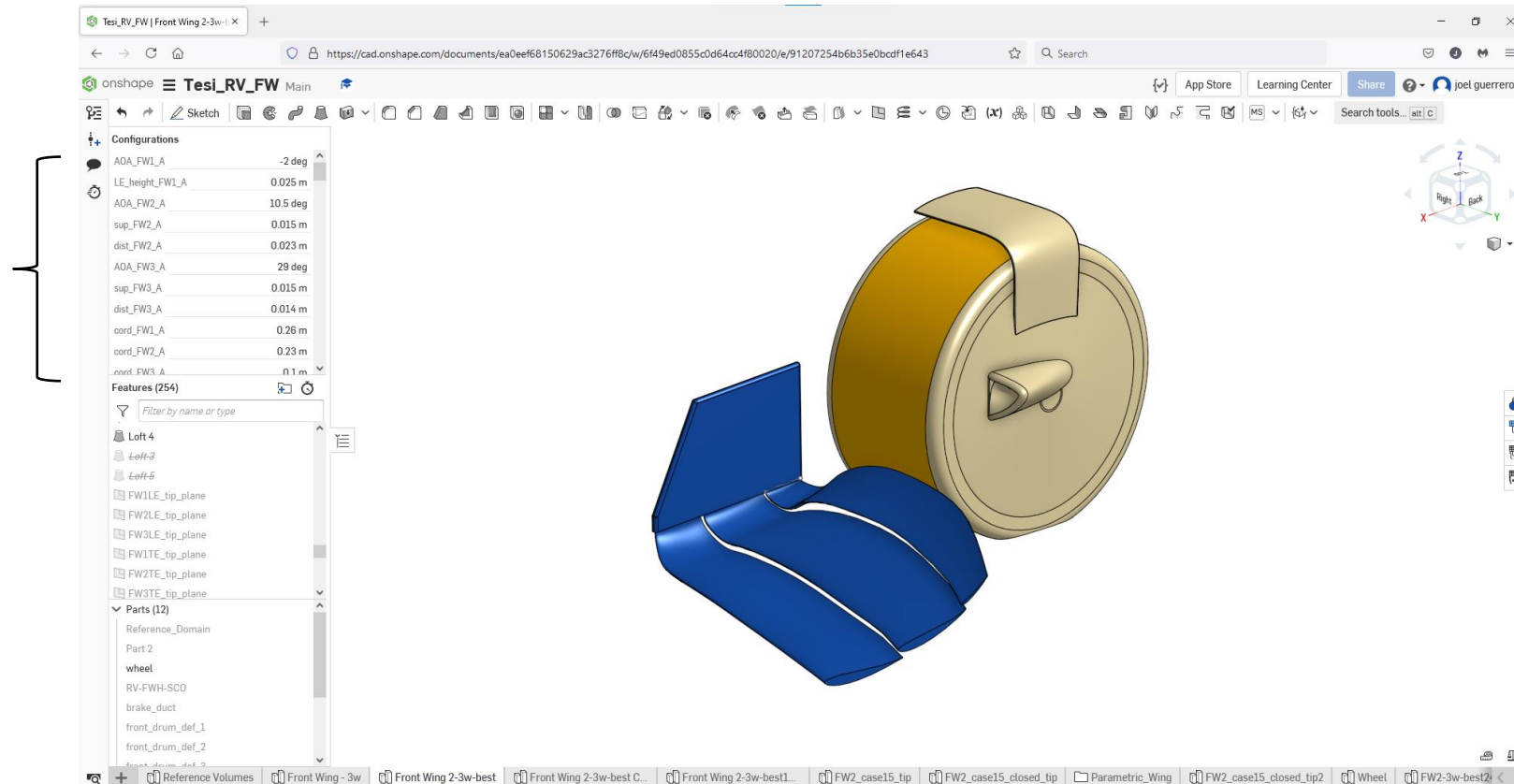
CFD optimization sample cases:

Optimization of the front wing of an F1 car

CFD optimization sample cases

- Optimization of the front wing of an F1 car – Geometry based on the FIA 2022 technical regulations.
 - The main goal is to maximize the downforce.
 - A secondary goal is to modify the concentrated vortex generated by the wheel-wing configuration.
 - We first conducted a DSE study and then we fine tuned the geometry using the adjoint.

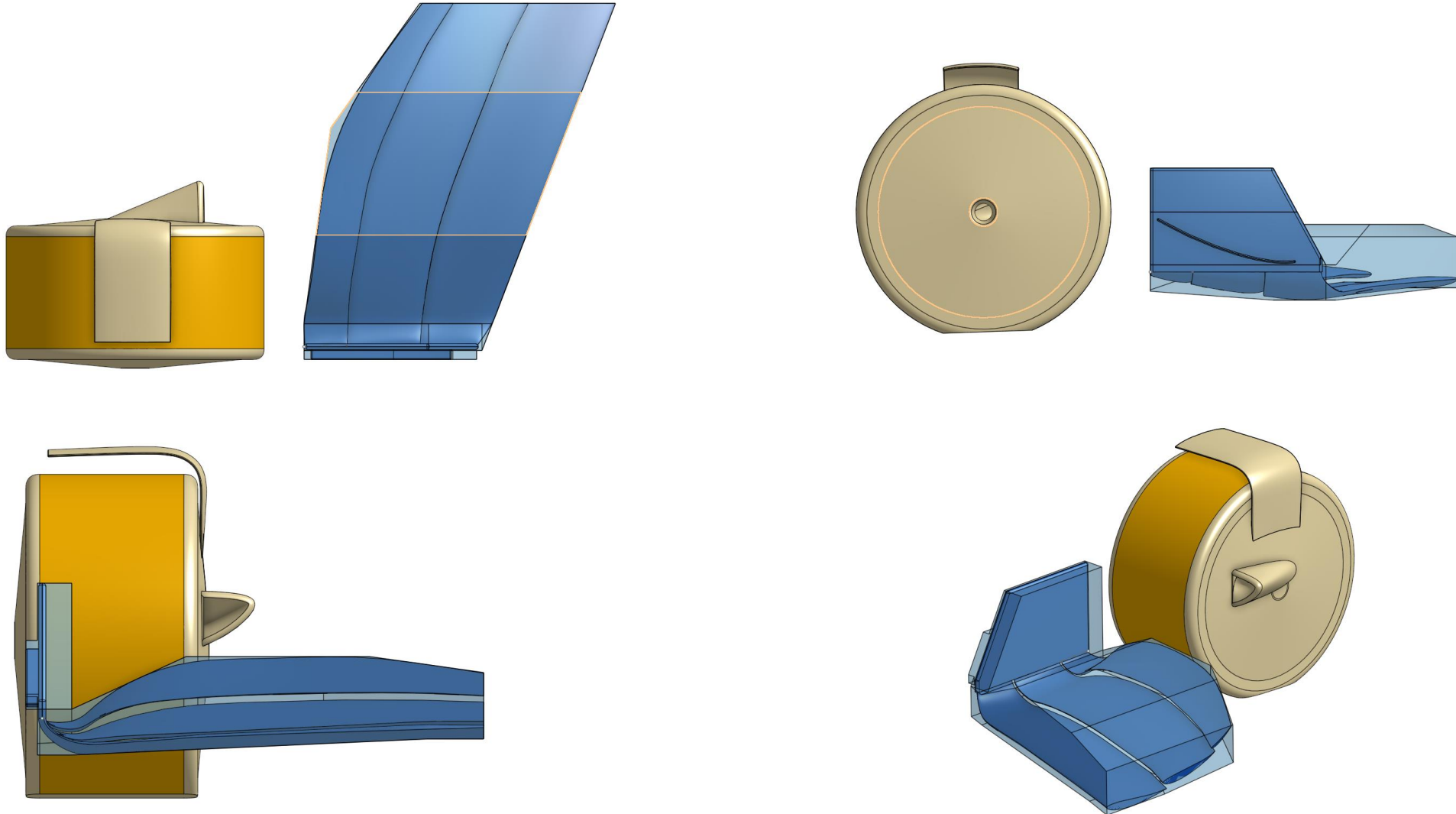
- There 100 parametrical variables.
- For the DSE experiment we only modified 4 variables (incidence angle of the airfoil)



* We would like to thank Marco Giachi for the enlightening discussions and valuable suggestions. If you want to know more about Marco's work, look for his book: Capire la Formula 1. I segreti della sua evoluzione dagli anni '60 a oggi. Minerva Edizioni (Bologna); 2° edizione (30 ottobre 2018).

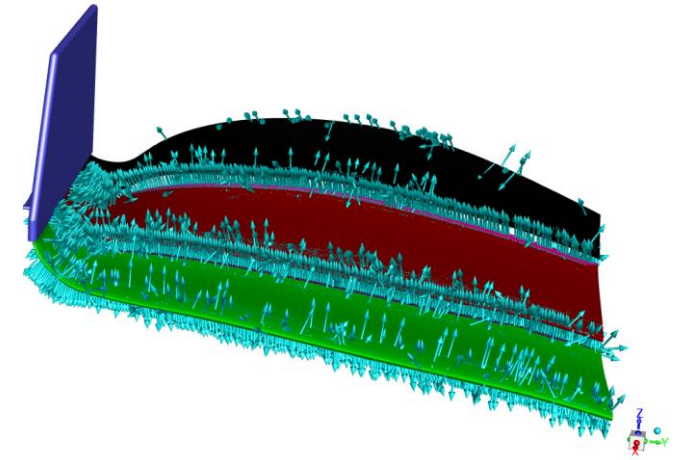
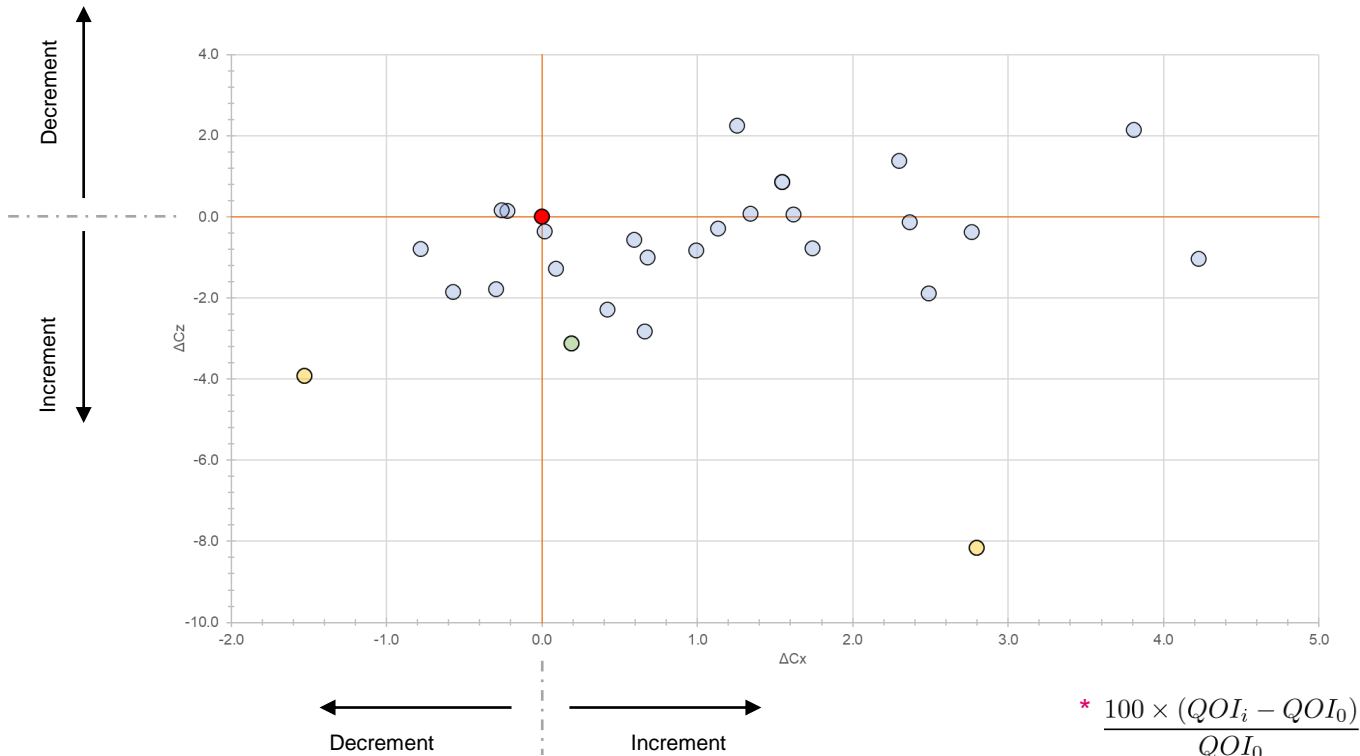
CFD optimization sample cases

- The transparent box around the front wing represents the feasibility region (the new geometry cannot go outside this box).
- Besides this constraint, there are many other constraints to respect according to the FIA 2022 technical regulations.

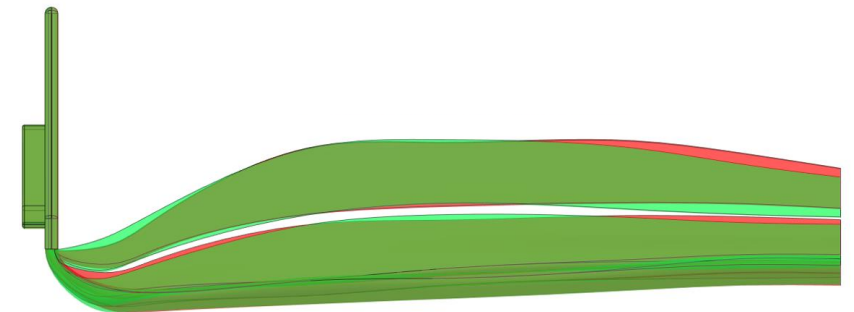


CFD optimization sample cases

- Quantitative results.
- Downforce coefficient ($C_z = -C_L$) variation against drag coefficient (C_x) variation.
- The variation (or the percentage error*) is computed in reference to case zero (red circle in the figure).
- In the figure, the green circle represents the best case from the DSE, the orange circles represent adjoint solutions, and the blue circles represent DSE simulations.



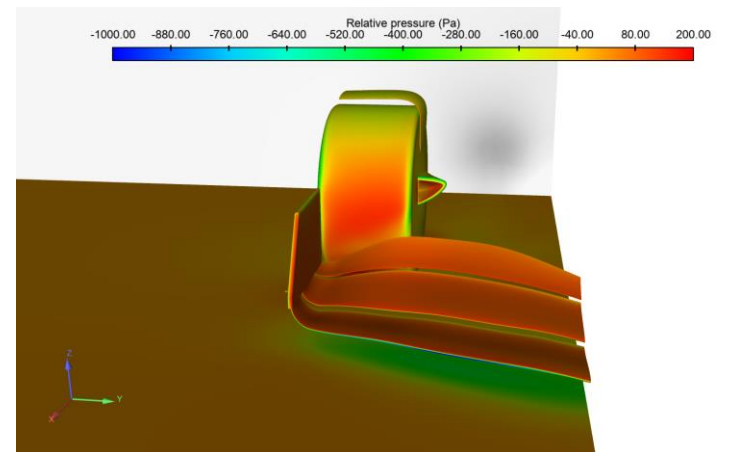
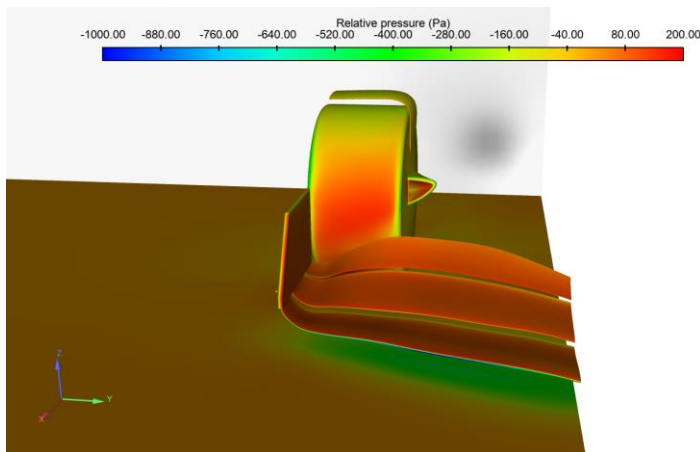
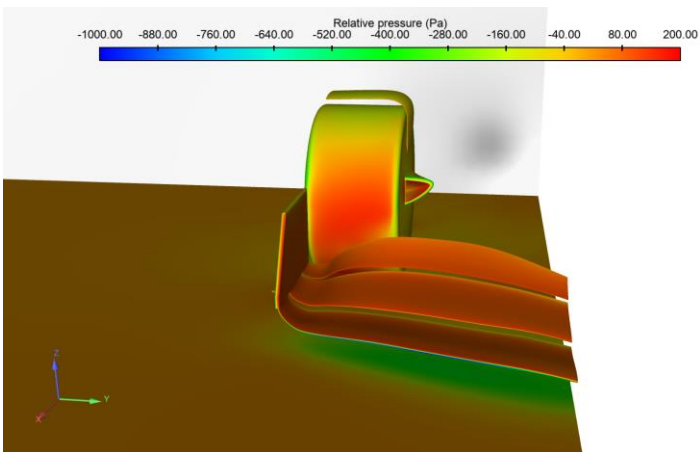
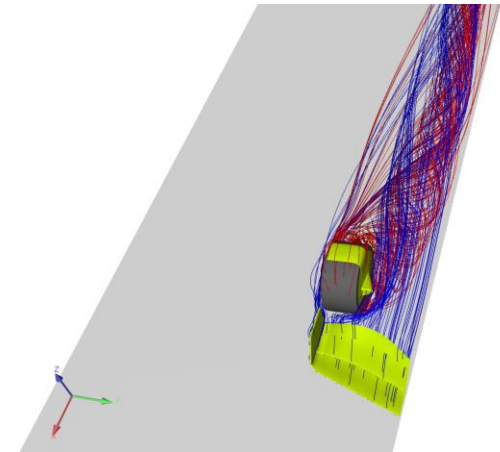
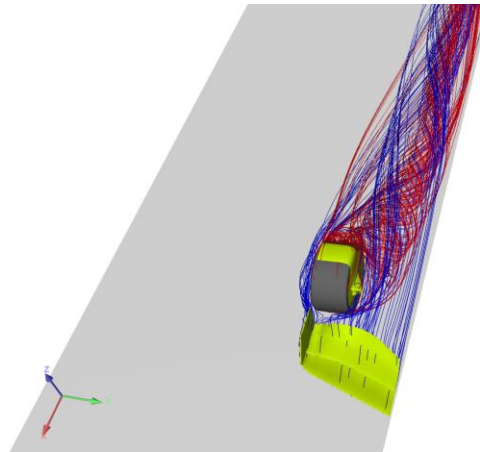
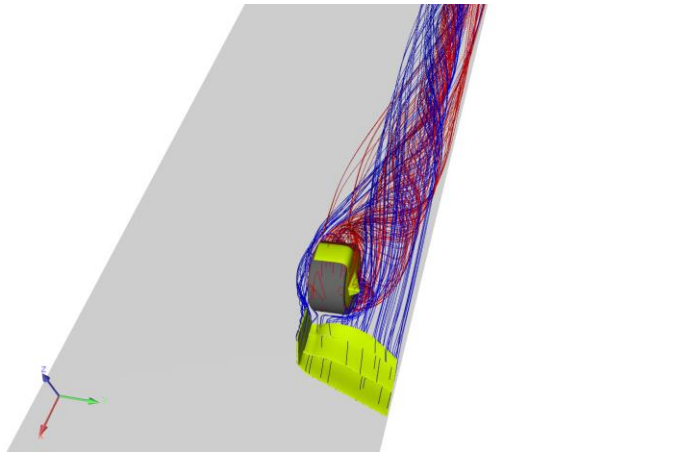
Design velocity vector



Case base
Best case from DSE

CFD optimization sample cases

- Qualitative results.
- Pressure contours of the walls – Streamlines released from the wheel-wing configuration – Location of the vortex core in a plane behind the wheel-wing configuration.



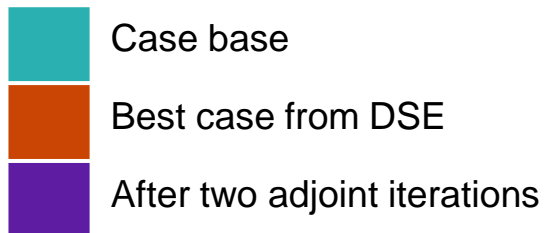
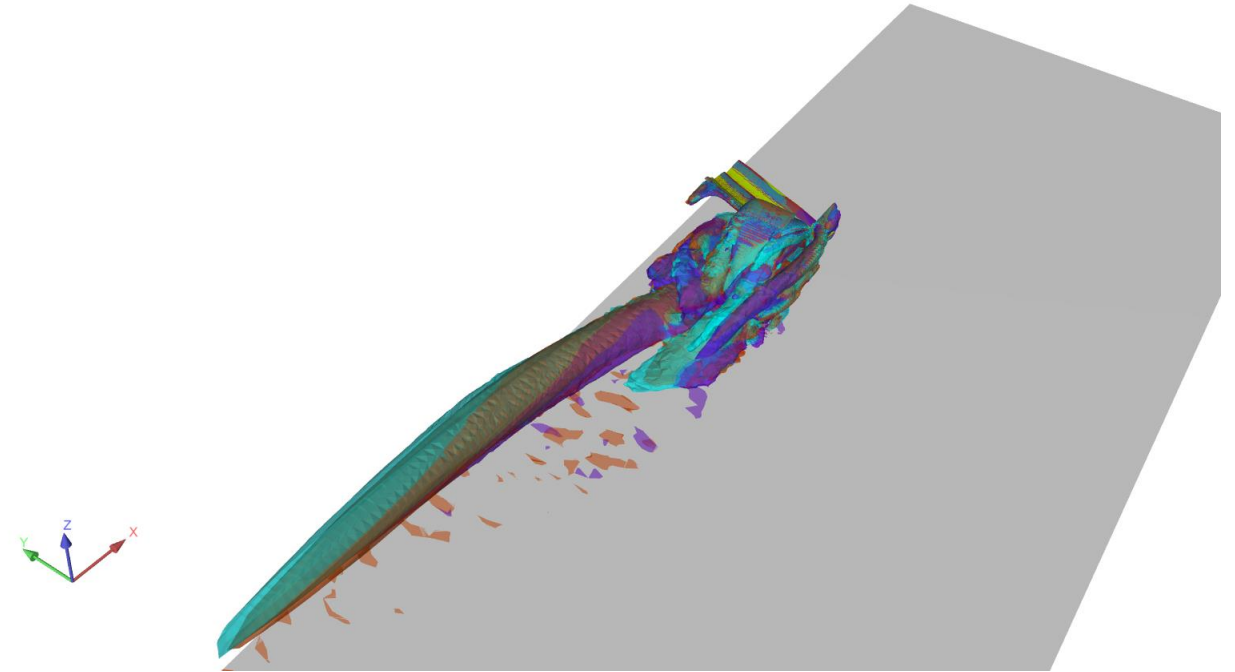
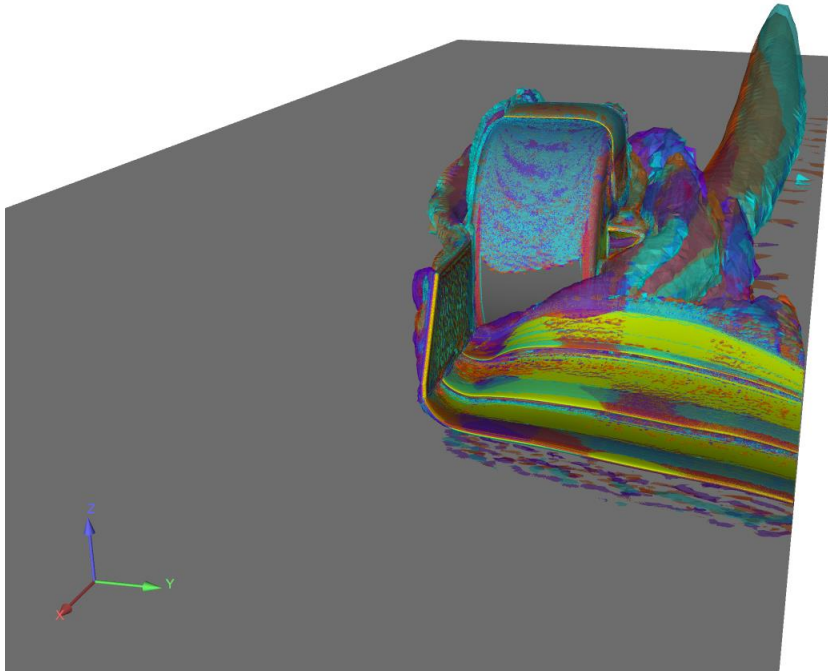
Case base

Best case from DSE

After two adjoint iterations

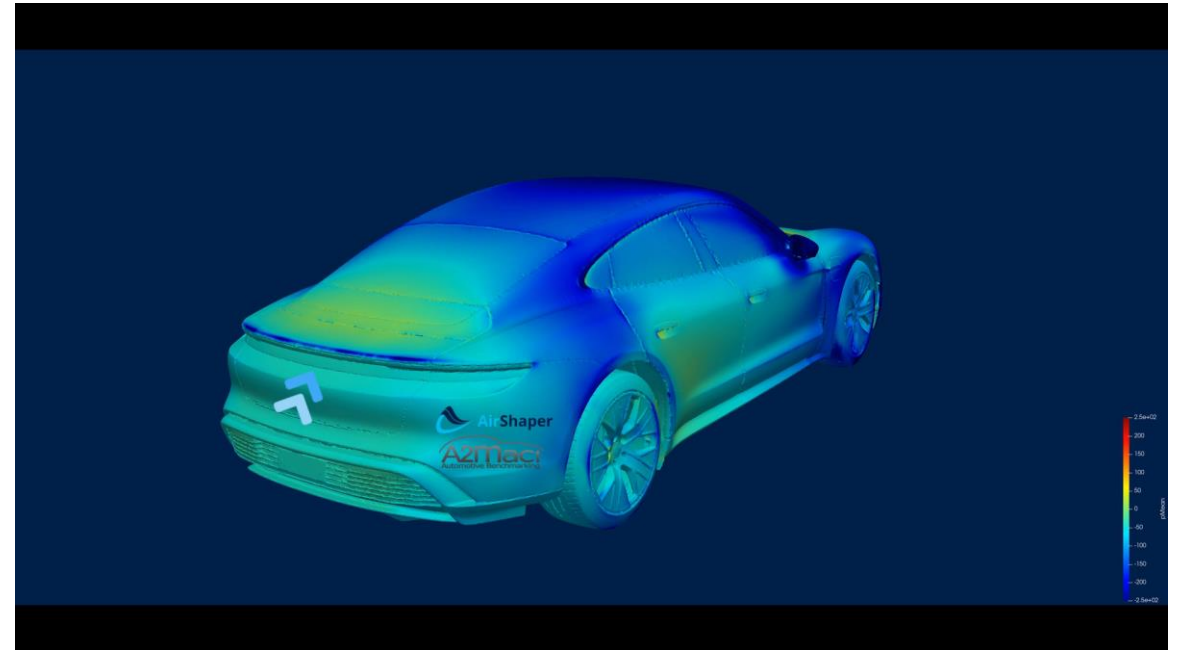
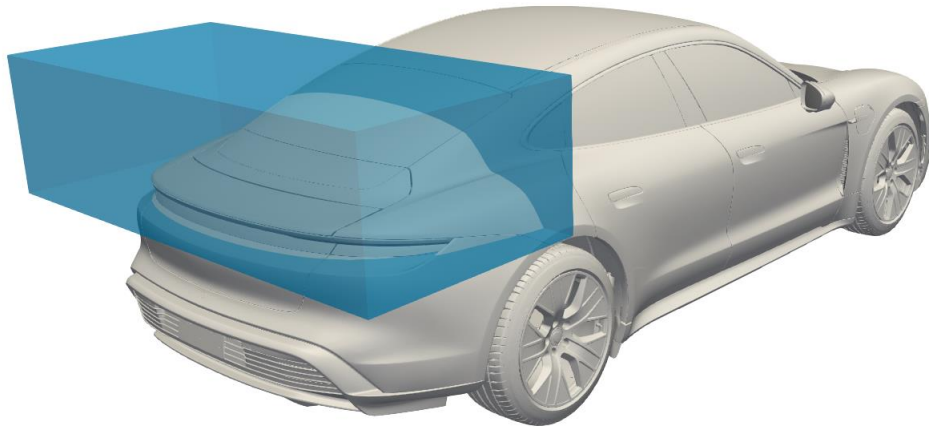
CFD optimization sample cases

- Qualitative results.
- Visualization of the vortical structures using the Q-criterion.



CFD optimization sample cases

- It is worth mentioning that we did not use OpenFOAM's adjoint solver.
- At the moment, we are collaborating with Airshaper* to obtain a solution with OpenFOAM's adjoint solver and compare results and computational efficiency.



<http://www.wolfdynamics.com/training/opt/media1.mp4>

* <https://airshaper.com/>



5. Live demonstration

Be collaborative, be innovative,

be cloud



wolfdynamics
multiphysics simulations,
optimization & data analytics

Let's connect



guerrero@wolfdynamics.com



www.wolfdynamics.com