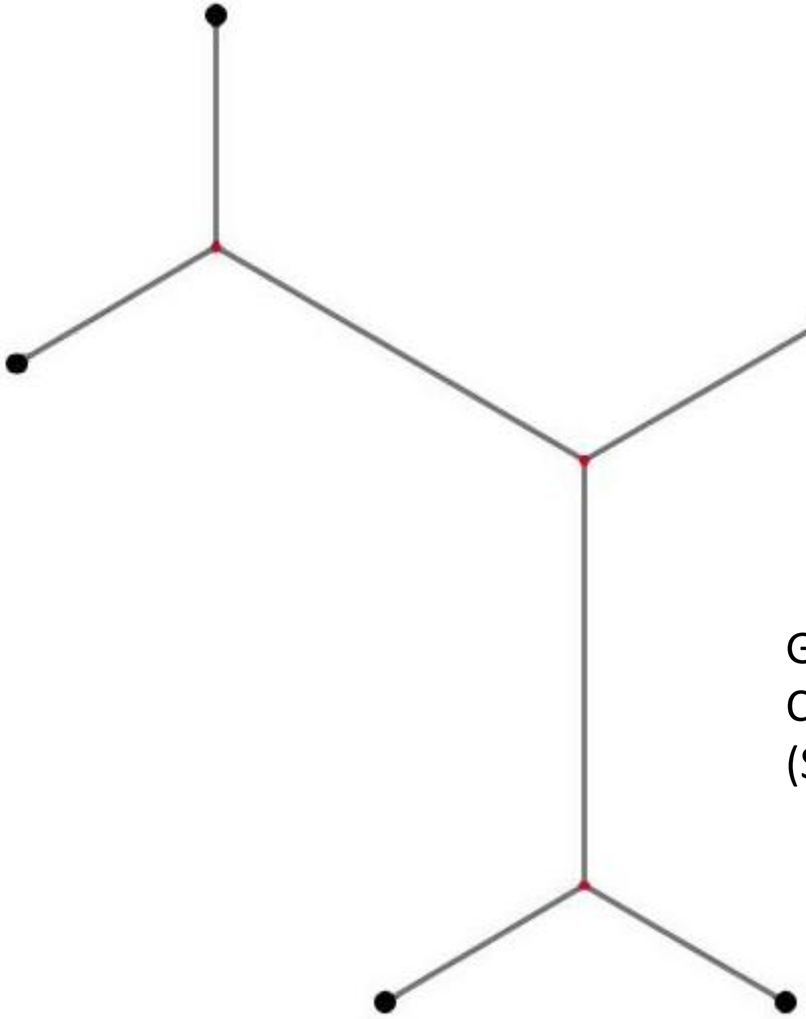
Design Optimization Problems

Michael Emmerich

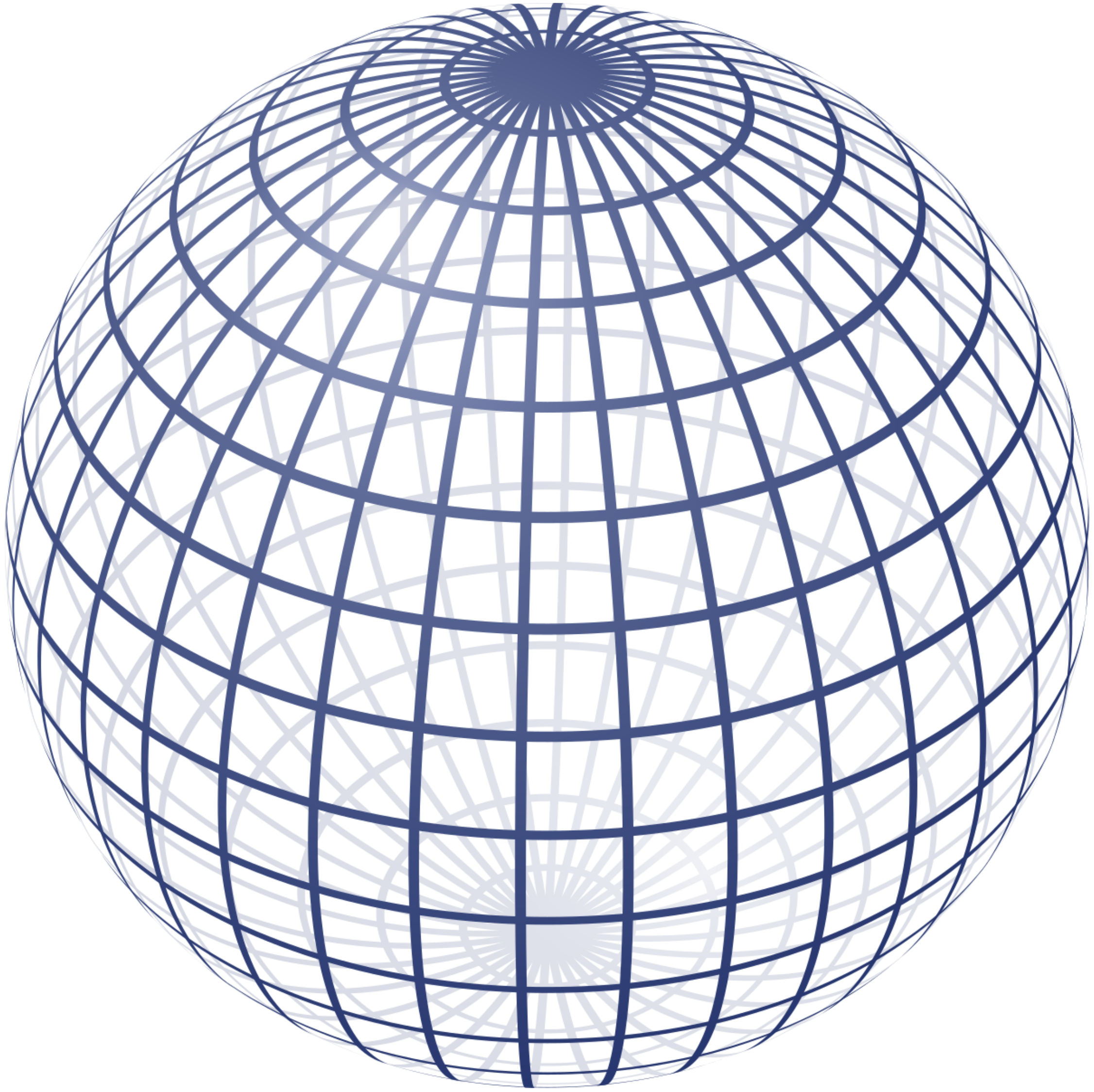
Learning Goals

- Platon’s dream: What are ideal solutions to design problems? Where/why/in which world do ideal shapes exist?
- Design as a discipline
- Understanding how to couple simulators and deal with black-box constraints; What are penalty functions?
- Parameterization: How to encode geometrical shapes by means of vectors or point sets
- How to gain insight into a design problem and its ideal solutions by means of optimization; what is $innovization$?

Perfect Structures

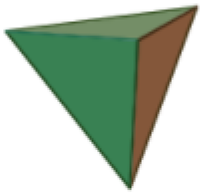
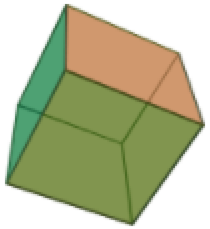
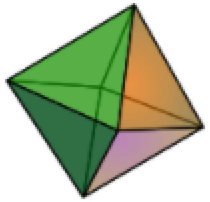
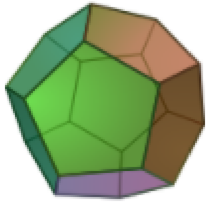
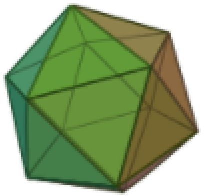


Given points
Connect with min path
(Steiner tree)



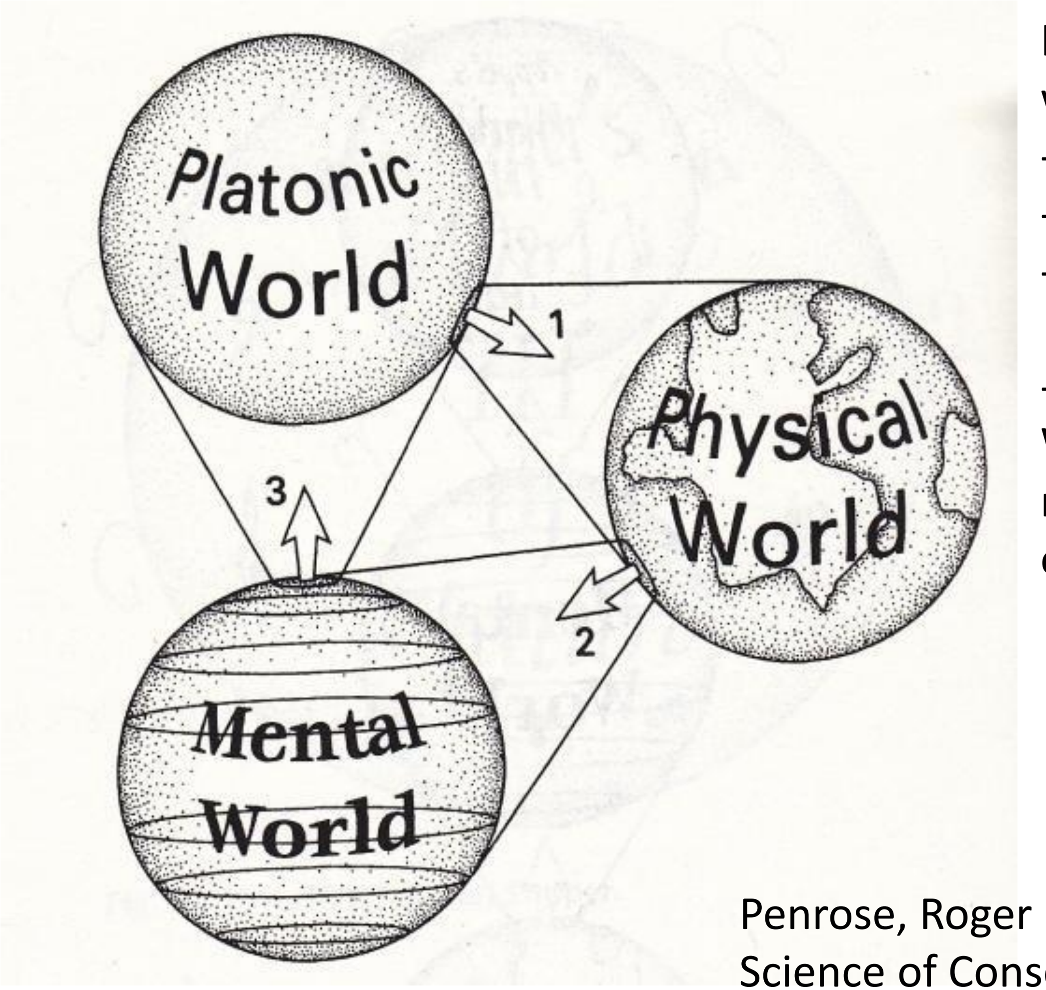
Min Surface, Volume = const

Platonic bodies – 'Ideal' shapes

Tetrahedron	Cube	Octahedron	Dodecahedron	Icosahedron
Four faces	Six faces	Eight faces	Twelve faces	Twenty faces
				
(Animation)	(Animation)	(Animation)	(Animation)	(Animation)
(3D model)	(3D model)	(3D model)	(3D model)	(3D model)

Shapes of maximally even (equal) surfaces at their boundary
https://en.wikipedia.org/wiki/Platonic_solid

Penrose: Three worlds, or where may ideal structures exist and who can perceive them?



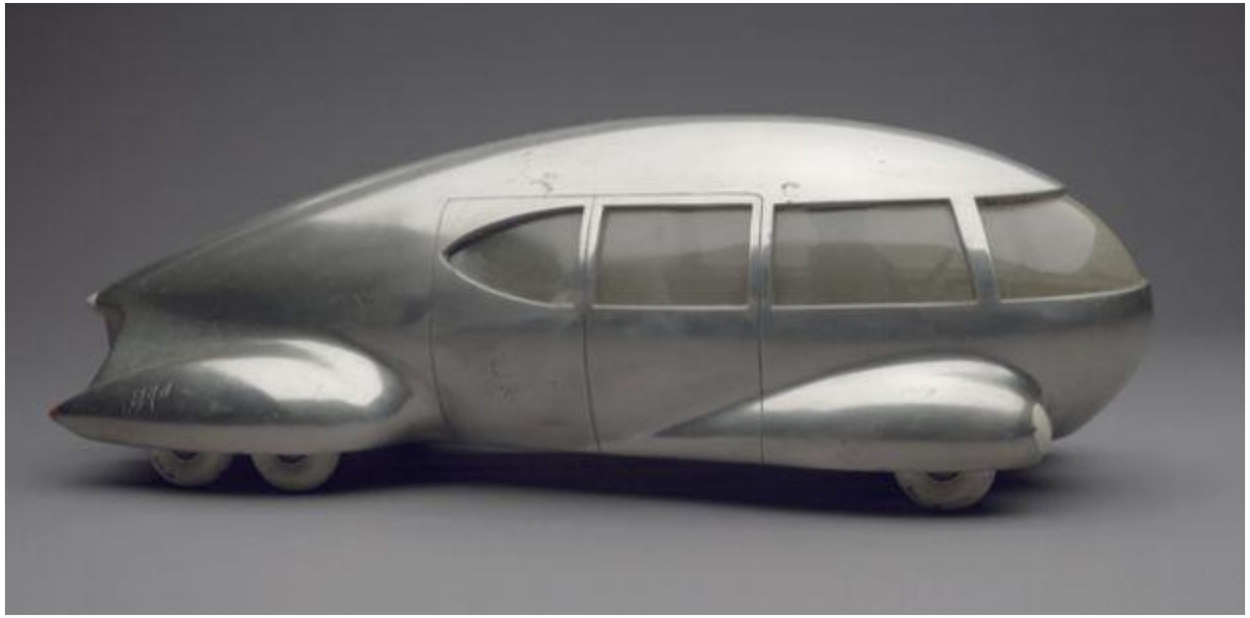
It remains a deep mystery how these worlds were/are created in the first place …

- How new natural laws can appear?
- How mathematical truths came into being?
- How new conscious beings with insight can start to exist? ….
- Is the three world model sound and complete?

We are not claiming answers for such deep questions merely looking at it here from the point of view of design optimization.

Penrose, Roger (1989). Shadows of the Mind: A Search for the Missing Science of Consciousness. Oxford University Press. p. 457

Perfect Designs, $c_w = \min$



Norman Bel Geddes, "Motor Car No. 9 (without tail fin)"

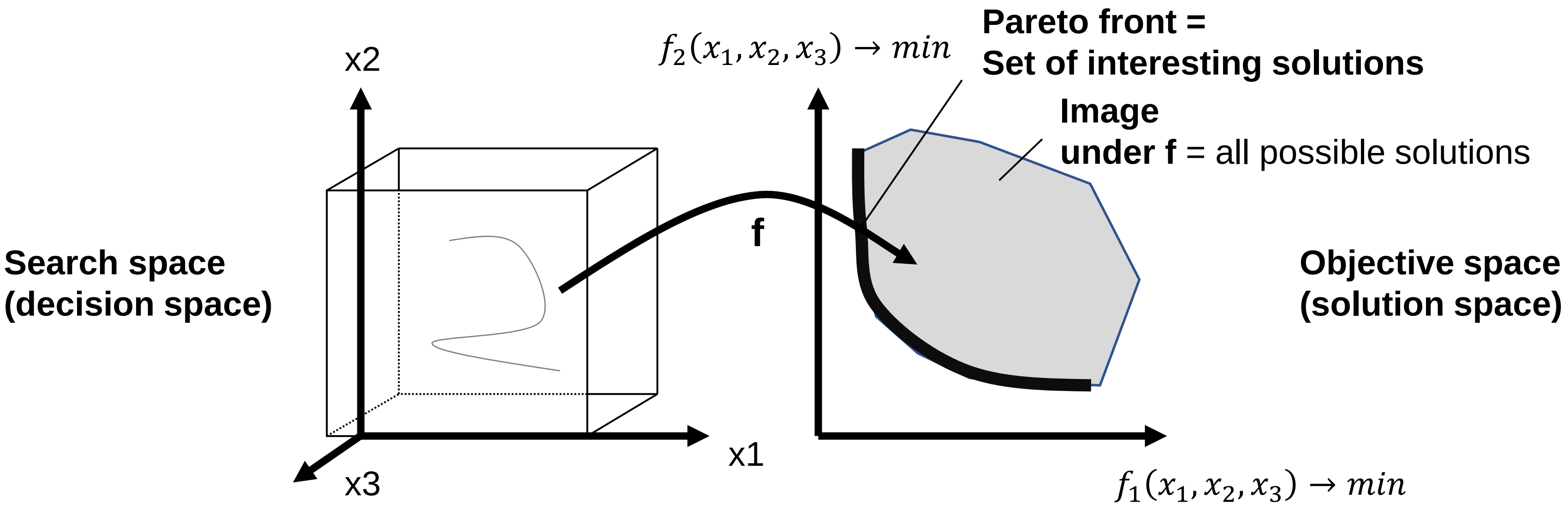
From: Horizons: Norman B. Geddes, Little Brown, Boston, 1932.

Aalto shape (beauty and aesthetics is hard to measure, but important in design …)

https://www.subpng.com/free-png/aalto-vase.html

Many people in Finland like this shape, perhaps it reminds them of a beautiful lake (a positive association) …. Often ‘natural shapes’ are perceived as beautiful. Think of examples of beautiful shapes (that many people like) in nature, architecture …. But it is hard to measure quantitatively. (perhaps by machine learning based ontraining data…)

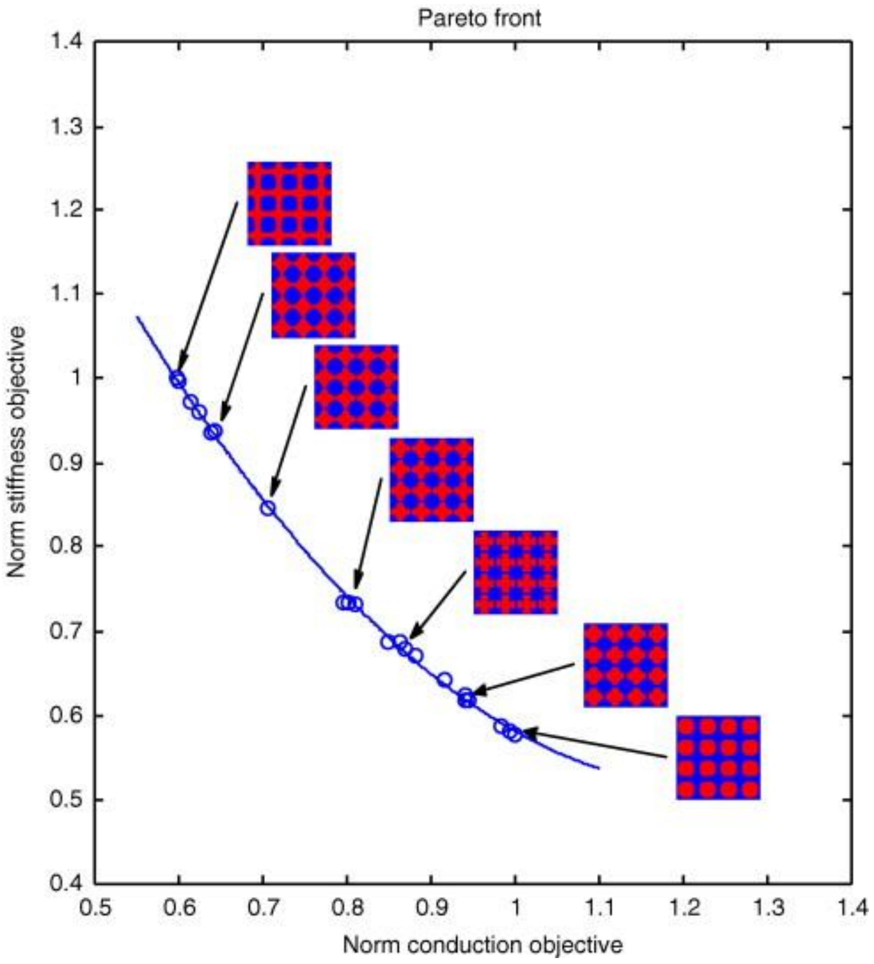
Pareto optimality



Recall a Pareto front. We find also here ideal stuctures, ideal in the sense of Pareto optimality. We can get insights from looking at solutions on the Pareto front.

Research questions

- Find ‘Pareto perfect’ structures:
    - Micro-
    - Macro-
- Efficiency
- Precision & Coverage



Niek de Kruijf, Shiwei Zhou, Qing Li, Yiu-Wing Mai,Topological design of structures and composite materials with multiobjectives, International Journal of Solids and Structures, Volume 44, Issues 22-23, 2007, Pages 7092-7109

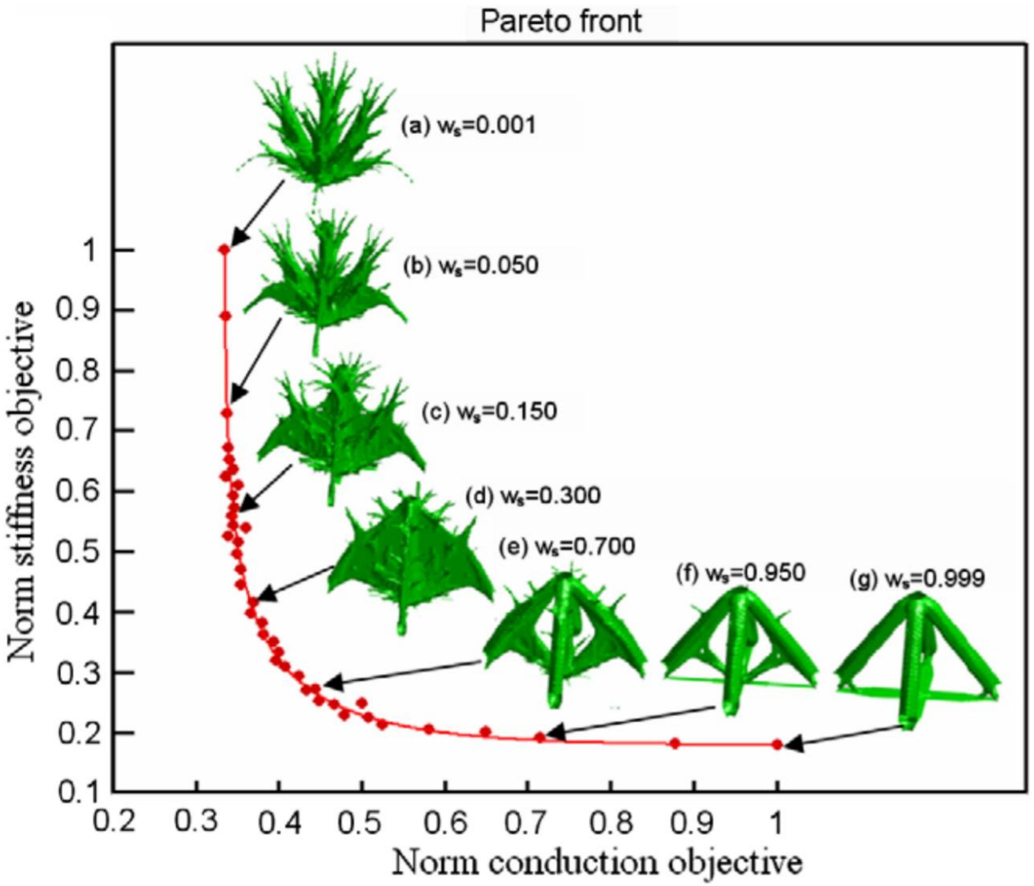
‘Pareto morphing’ along the Pareto front

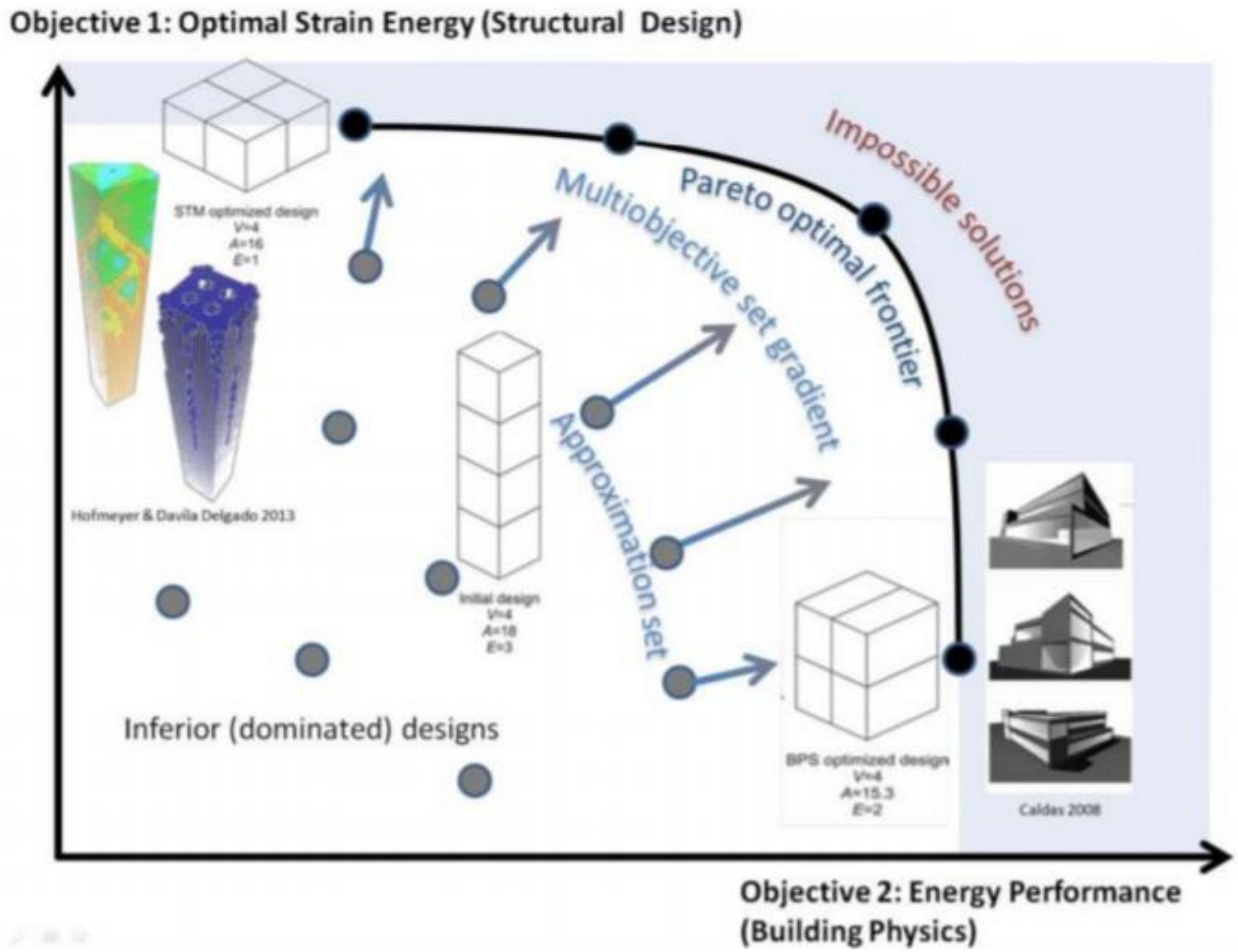
How do designs change along the Pareto front?

Innovization: Learing design principles from optimization:

Term coined by: K. Deb (US-Indian Engineer and Computer Scientist) S. Bandaru (Swedish-Indian Engineer and Computer Scientist)

Chen, Yuhang, Shiwei Zhou, and Qing Li. "Multiobjective topology optimization for finite periodic structures." *Computers & Structures* 88.11-12 (2010): 806-811.





Objective 1: Optimal Strain Energy (Structural Design)
STM optimized design
V=4
A=16
E=1
Hofmeyer & Davila Delgado 2013
Impossible solutions
Pareto optimal frontier
Multiobjective set gradient
Approximation set
Initial design
V=4
A=18
E=3
Inferior (dominated) designs
BPS optimized design
V=4
A=15.3
E=2
Caldas 2008
Objective 2: Energy Performance
(Building Physics)

Question

- How can we express shapes by means of decision variables?
- How can we make sure that constraints are kept?

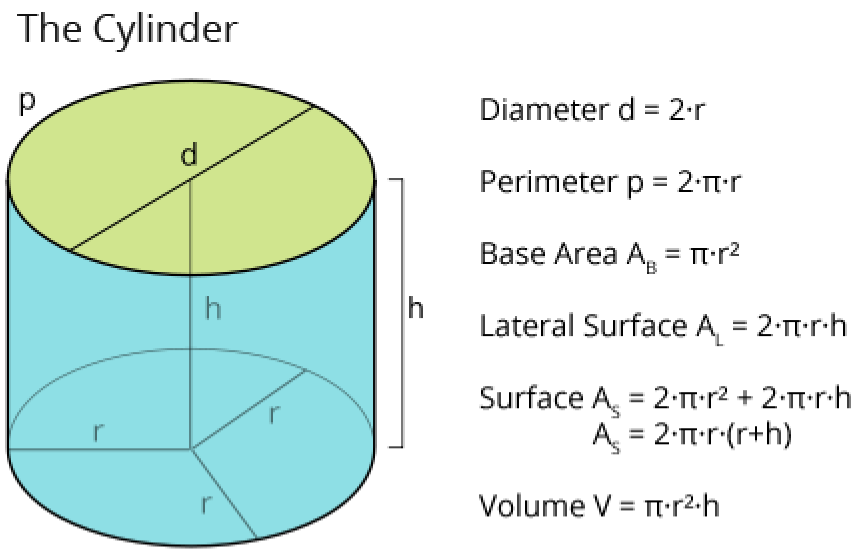
Example: Cylinder (tin)

Volume → max

Surface → min

Parameters: r, h

Constraints: r > 0, h > 0



Solution by substitution method and epsilon constraint

$$V(r,h) = \pi r^2 h = \epsilon \Rightarrow h = \frac{\epsilon}{r^2}$$

$$A(r,h) = 2\pi r^2 + 2\pi r h$$

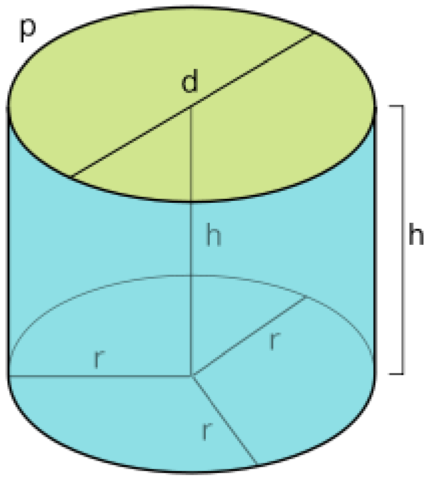
$$= 2\pi r^2 + \frac{2\pi r}{\pi r^2} = 2\pi r^2 + \frac{2}{r} \rightarrow min$$

$$\nabla A(r) = \frac{dA(r)}{dr} = 4\pi r - \frac{2\epsilon}{r^2} = 0$$

$$\Leftrightarrow r^3 = \frac{2\epsilon}{4\pi} \Leftrightarrow r(\epsilon) = \sqrt[3]{\frac{\epsilon}{2\pi}}$$

$$\nabla^2 A(r) = 4\pi + \frac{2\epsilon}{r^3} > 0 \text{ (sufficient condition)}$$

The Cylinder



Efficient Set:

$$\mathrm{X_e} = \{(r(\epsilon), h(\epsilon)) | \epsilon \in (0, \infty], r(\epsilon) = \sqrt[3]{\frac{\epsilon}{2\pi}}, h(\epsilon) = \frac{\epsilon}{\pi r^2})\}$$

Pareto front:

$$Y_{nd} = \{(A\big(r(\epsilon), h(\epsilon)\big), V\big(r(\epsilon), h(\epsilon)\big) | \epsilon \in (0, \infty]\}$$

Basic strategy in Black-box optimization

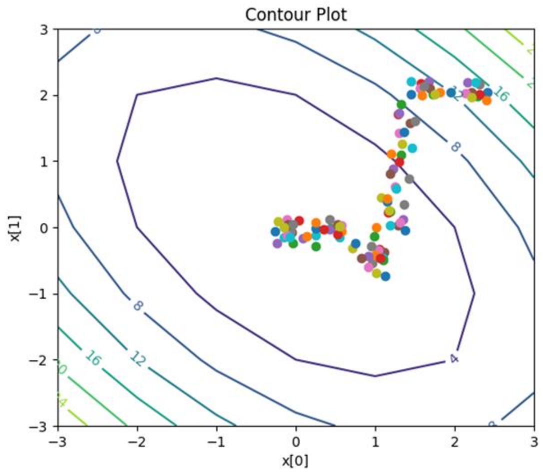
Black-box Optimierungs-software

Input Parameters x(t)

Simulator/Evaluator

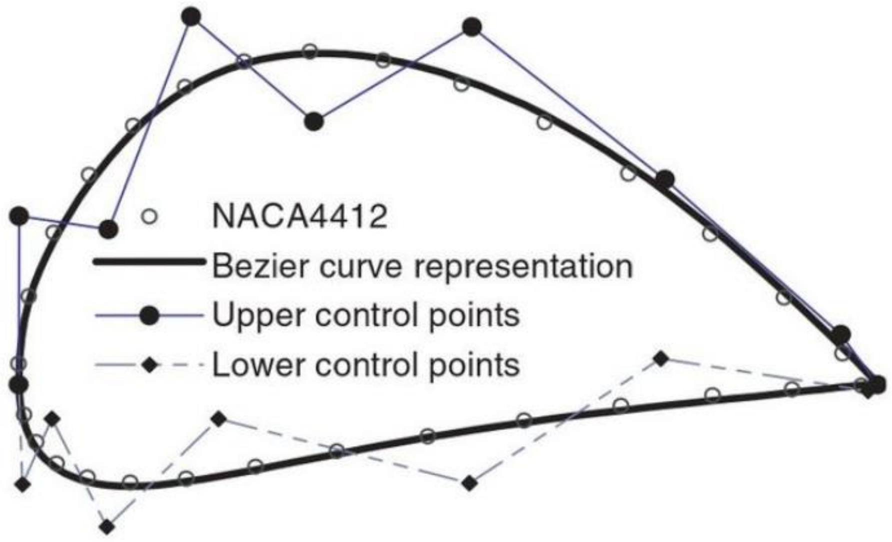
Zielfunktionswerte, Restriktionsverletzungen
f(x(t))+penalty(r(x(t))

1. Stochastic Hillclimbing
2. Gradient Descent
3. Newton Method
4. Simulated Annealing
5. Evolutionary Algorithm
6. Bayesian Optimization
7. Etc.



Design Parameterization

- Design Parameterization is the problem of describing a geometrical shape by means of continuous parameter vectors
- Examples: Bezier points, superstructures



Figure

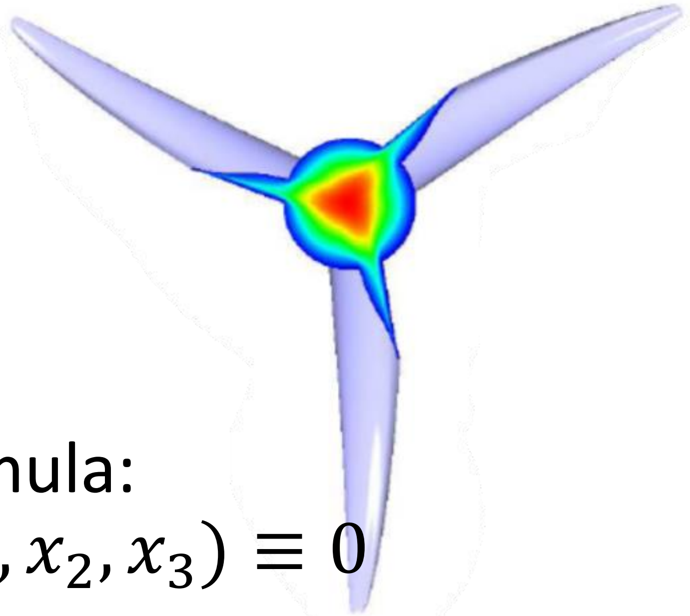
Caption

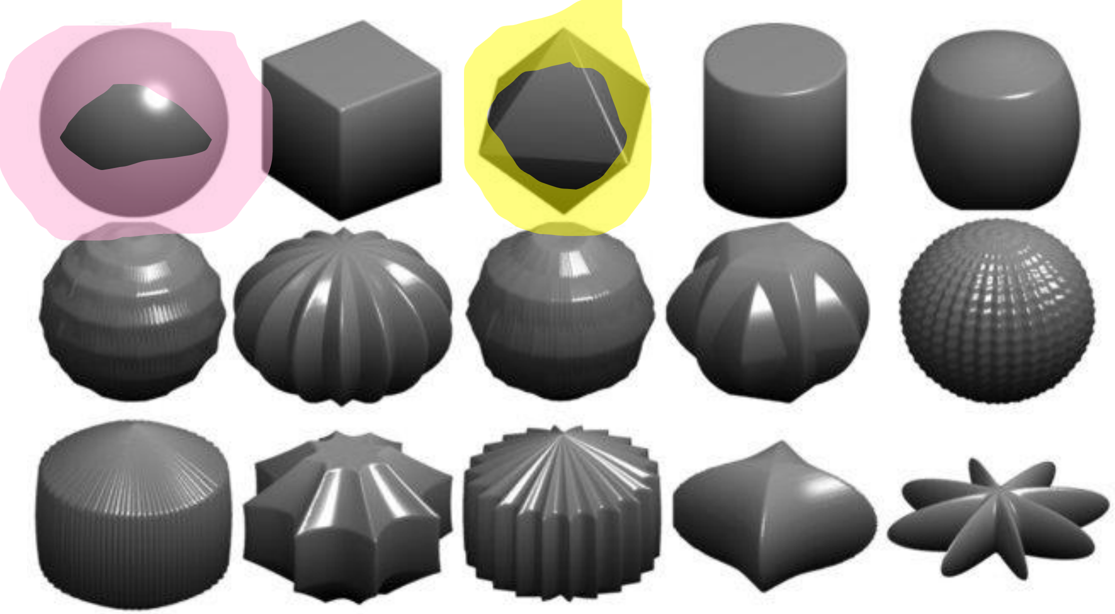
Fig. 1.8: Bezier curves representation of NACA4412 airfoil [36]

This figure was uploaded by Stavros N. Leloudas
Content may be subject to copyright.

Source: https://arxiv.org/pdf/1204.4107.pdf

Parameterization
Gielis' superformula





$$r(\varphi) = \left(\left| \frac{\cos\left(\frac{m\varphi}{4}\right)}{a} \right|^{n_2} + \left| \frac{\sin\left(\frac{m\varphi}{4}\right)}{b} \right|^{n_3} \right)^{-\frac{1}{n_1}}.$$

- The superformula: $f(p_1, \ldots, p_n, x_1, x_2, x_3) \equiv 0$
- $x_1, \ldots x_n$: variables
- $p_1, \ldots, p_n$: parameters
- Example

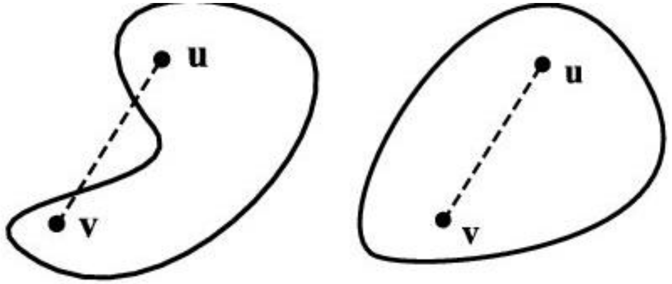
$$f(x_1, x_2, x_3) = \sum |x_i|^{p_i}$$
$$p_i = 1, i = 1, \ldots, 3$$
$$p_i = 2, i = 1, \ldots, 3$$

Convexity

Idea
Represent buildings as convex shapes

'Non-convex' roof
Of Station Oriente
in Lisbon

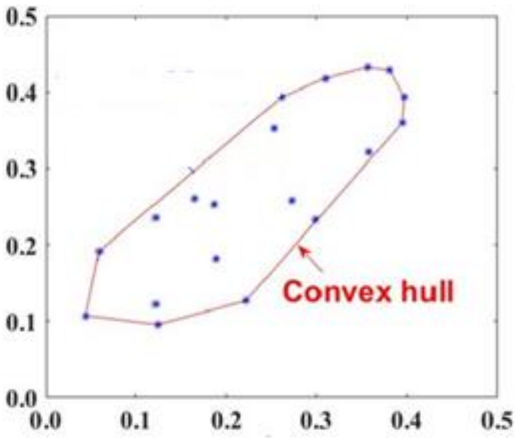
- A set $S \subseteq \mathbb{R}^d$ is called a convex set if it for any two points, $u \in S$ and $\mathrm{v} \in S$ the line-segment connecting u and v is fully contained in S.
- These shapes have no-cavities (this is why buildings are often convex, excepting those with 'swimming pools' on the roof)

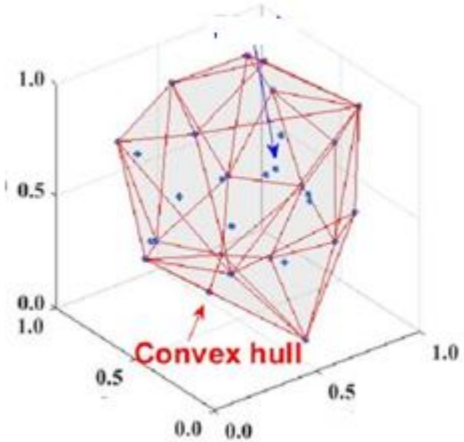


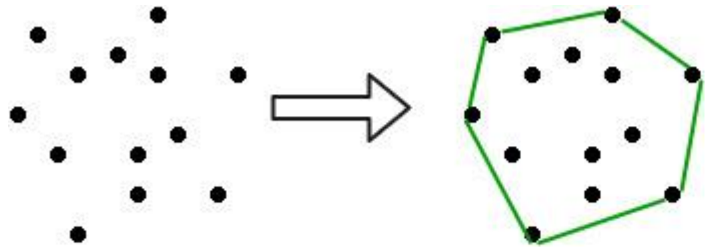
Donald Y. Tong; https://commons.wikimedia.org/wiki/File:(Cologne)_Central_Railway_Station,_Germany.jpg

https://commons.wikimedia.org/wiki/File:Oriente_Station_Lisboa_roof.jpg; Gerhard Missbach, Public domain, via Wikimedia Commons

Convex hull representation

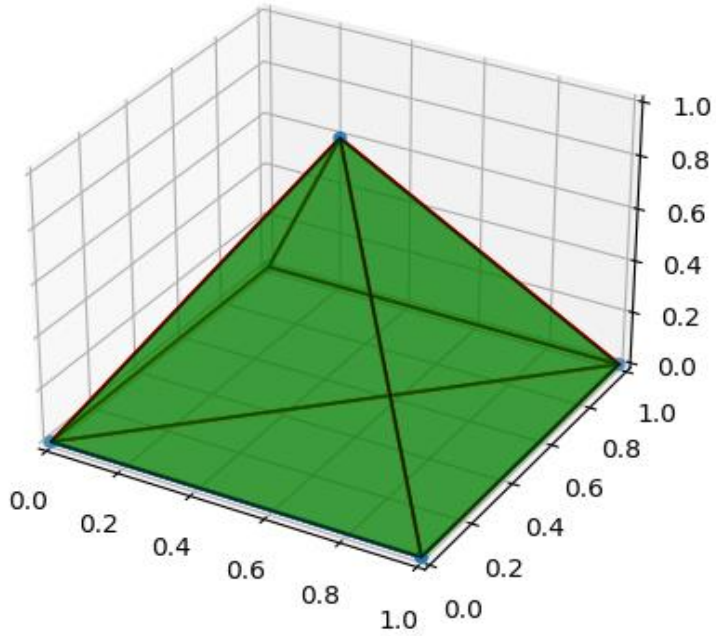


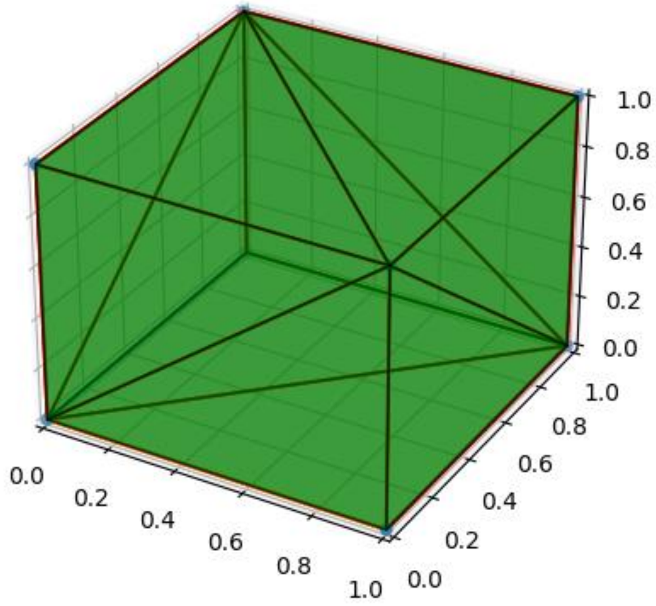




- The convex hull is the smallest convex set that contains all points
- In 2-D you can imagine a rubber-band around the set
- Convex hulls can be used to represent the set of all convex shapes by means of point sets
- ‘Active’ points form corners of this sets. Inactive points are redundant and can be removed.

Design optimization – 3D Shapes



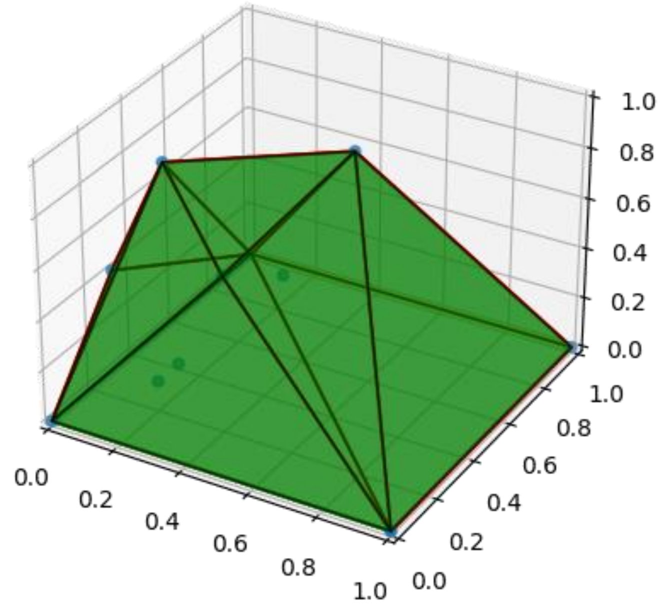


```python
# Pyramid example
print("Making a pyramid")
# Define the points first
pyramid_points = np.array([
    [0,0,0], [1,0,0], [0,1,0], [1,1,0], # floor corners
    [.5, .5, 1] # pyramidion / capstoneS
])
```

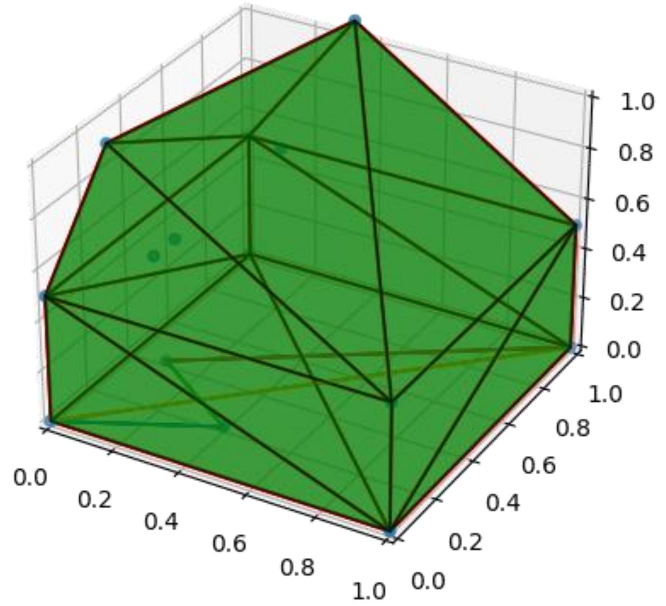
```python
# Box example:
print("Making a box")
# Define the points
box_points = np.array([
    [0,0,0], [1,0,0], [0,1,0], [1,1,0], # floor corners
    [0,0,1], [1,0,1], [0,1,1], [1,1,1] # Ceiling/roof corners
])
```

Walls, floors, and roofs

```
random_points = np.array([
    [0,0,0], [0,1,0],[1,1,0], [1,0,0], # floor
    [0, 0.3, 0.4+0.5], [0.4, 0.2, 0.6+0.5], [0.1, 0.4, 0.8],
    [0.4, 0.5, 0.4], [.5, 0.7, 0.8],
    [.2, 0.3, 0.1], [.2, 0.2, 0.1]
])
box3 = Tent(random_points)
box3.plot()
```

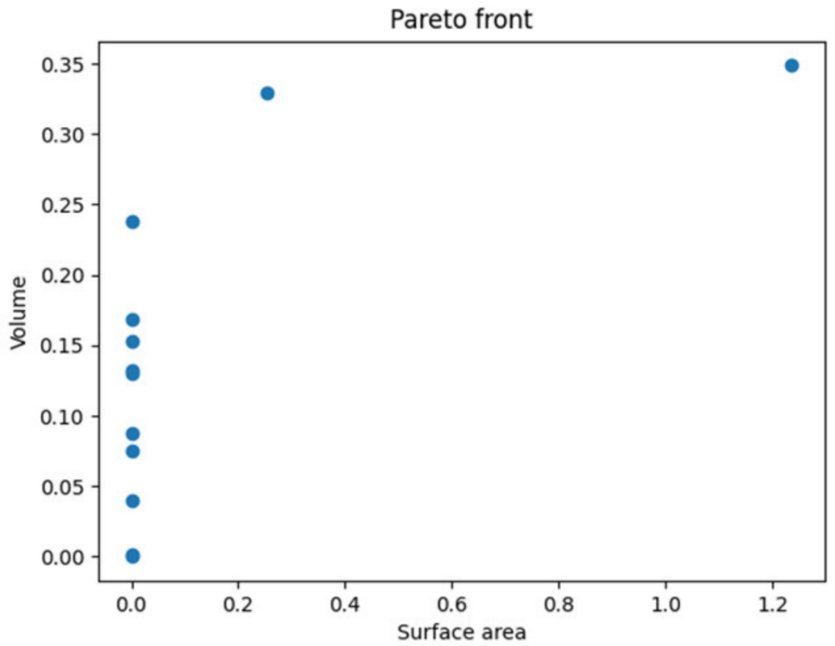


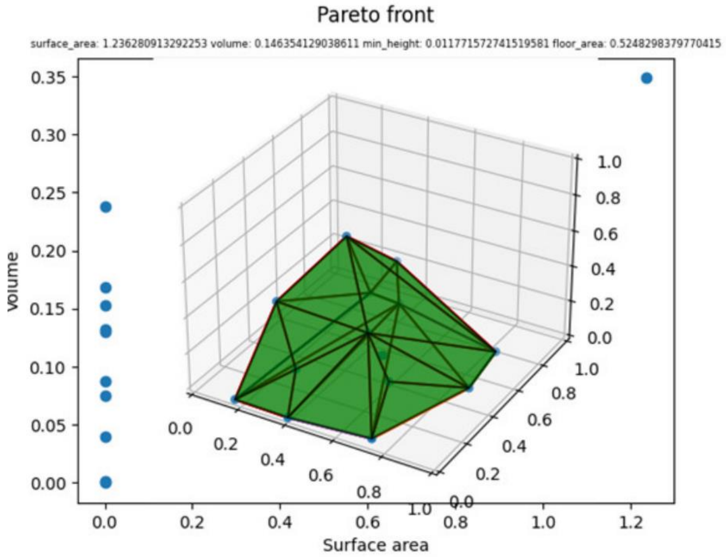

```
random_points = np.array([
    [0,0,0], [0,1,0],[1,1,0], [1,0,0], # floor
    [0, 0, 0.5], [0, 1, 0.5], [1, 1, 0.5],[1, 0, 0.5],  # floor+0.5
    [0, 0.3, 0.4+0.5], [0.4, 0.2, 0.6+0.5], [0.1, 0.4, 0.8+0.5],
    [0.4, 0.5, 0.4+0.5], [.5, 0.7, 0.8+0.5],
    [.2, 0.3, 0.1+0.5], [.2, 0.2, 0.1+0.5]
])
box3 = Tent(random_points)
box3.plot()
```

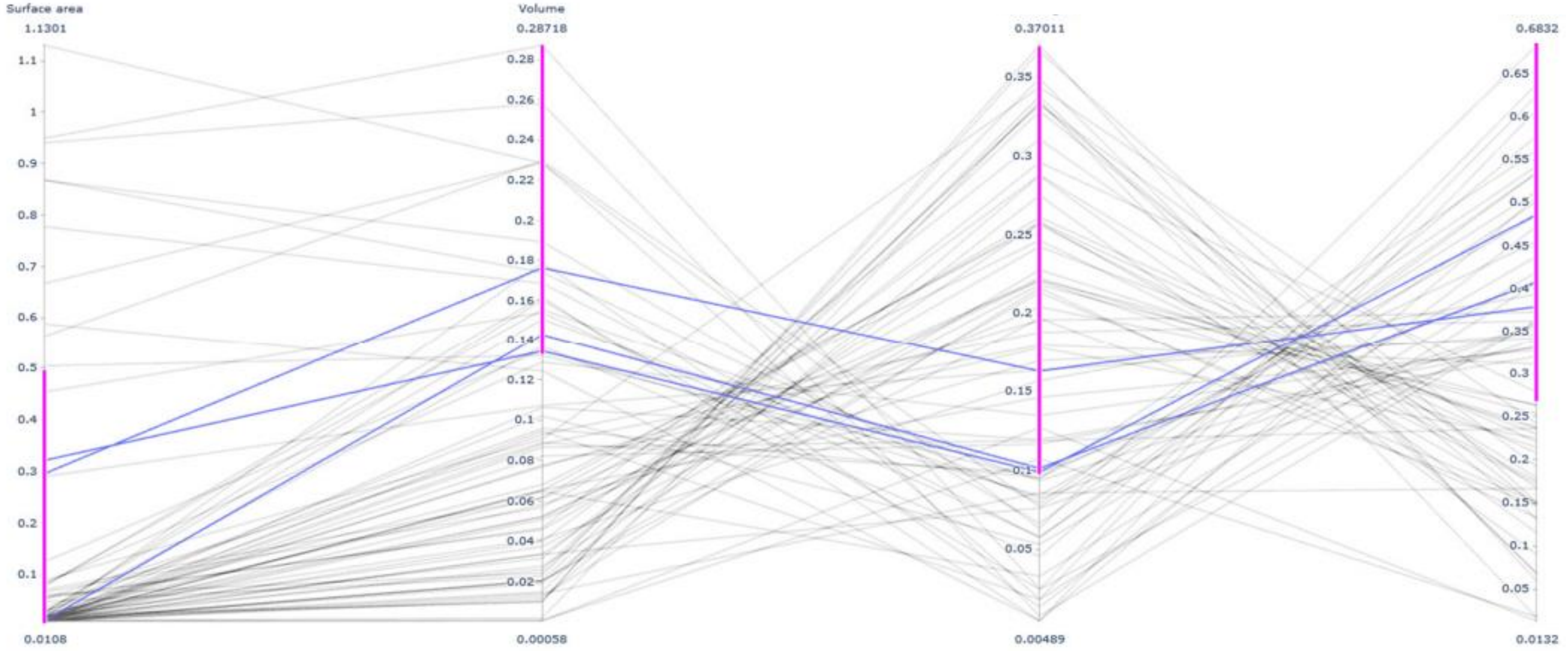


Optimizing Tents (homework)

- Maximize Volume
- Minimize Surface Area of Roof
- Maximize Height
- Maximize Floor Area







Walkthrough …. Tent example

Download geometry design problem, and add modules in your standard py directory; install desdeo_emo, desdeo_mcdm

```
from desdeo_mcdm.utilities.solvers import solve_pareto_front_representation
from desdeo_emo.EAs import NSGAIII
from modules.utils import save
from modules.GeometryDesign.problem import create_problem
import numpy as np
import warnings
warnings.filterwarnings("ignore") # ignore warnings :)
```

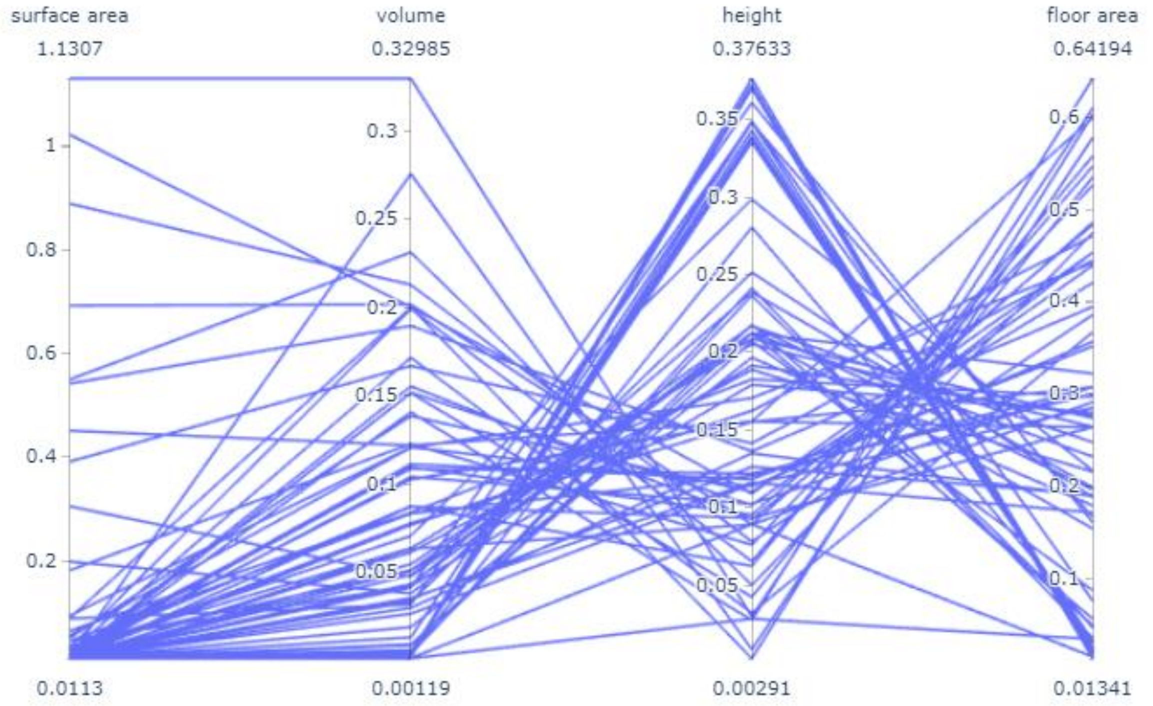
Creating the problem …

```python
# Creating geometry design problem : tent like buildings
# Which objectives do you wish to optimize
# surface area, volume, min height and floor area
obj = np.array([
    True, True, True, True, # Optimizing Surface area and min height and ignoring others,
])

# ideal and nadir in respective order
# ideal = 0, 1, 1, 1
# nadir = 5, 0, 0, 0


# Set constraint for objectives, [lower, upper]
# If no constraint then set it to None
# Each row represents a objective function in the same order as in obj_gd
# Notice that breaking constraints will result in a penalty and therefore we might get results that break the constraint
constraints = np.array([
    [0.2, None], # Surface area > 0.2
    [.5, .8], # .5 < volume < .8. Even though we're not optimizing volume, we can set a constraint on it
    [.4, None], #  min height > .4
    [None, 0.6], # floor area < .6
])

# How many 3d points should the hull be formed of
# more points => More complex problem : longer execution times
# Less points => More likely to fail in constructing the hull
variable_count = 15 # Around 15 - 25 seems to be good enough

# To create the problem we can call the gd_create method with the parameters defined earlier
# the pfront argument should be set to True if using the solve_pareto_front_representation method as it doesn't
# take account minimizing/maximizing. For everything else we can set it to False
# The method returns a MOProblem and a scalarmethod instance which can be passed to different Desdeo objects
problem, method = create_problem(variable_count , obj, constraints, pfront = True)
```

Optimization with NSGA_III algorithm

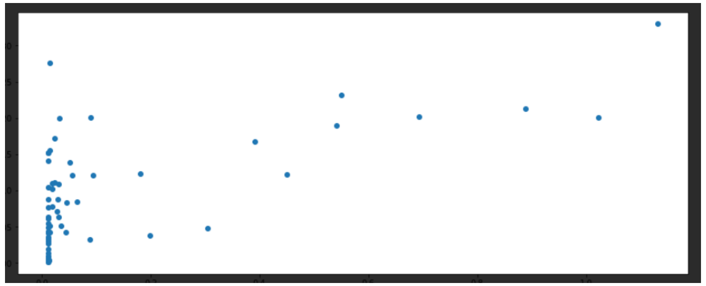
```
# Example on solving the pareto front using NSGA-III

evolver = NSGAIII(problem,
                  n_iterations=10,
                  n_gen_per_iter=100,
                  population_size=100)

while evolver.continue_evolution():
    evolver.iterate()

var, obj = evolver.end()

```

Plotting the results …



Parallel coordinates, concatenated decision and objective space

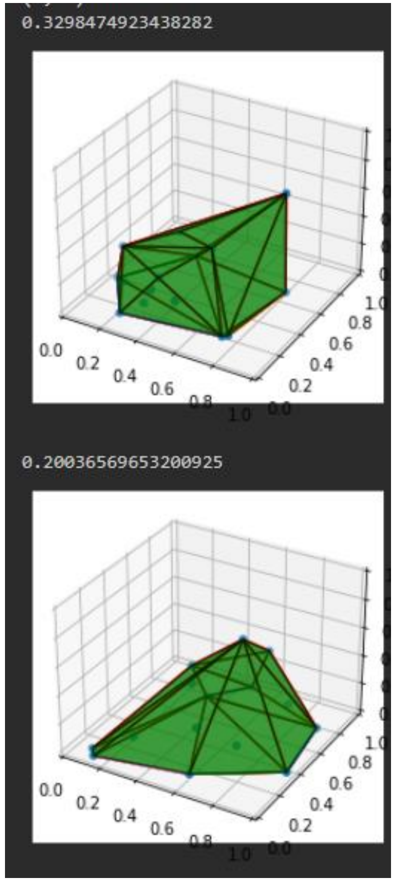
volume

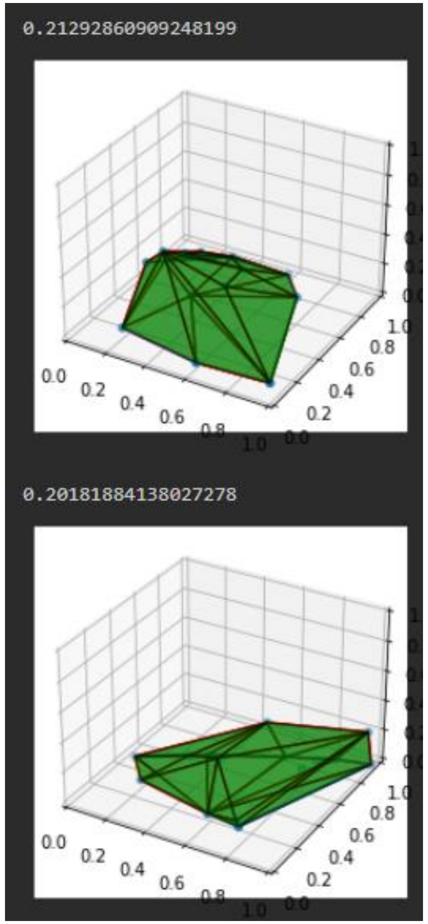


surface

```python

import plotly.graph_objects as go
fig2 = go.Figure(data=go.Scatter(x=obj[:,0],
                                 y=obj[:,1],
                                 mode="markers",
                                 marker_size=5))
fig2

#%%

from  modules.DataAndVisualization.vizualiser import (
    plot_scatter as scatter,
    plot_parallel as parallel,
)
axis_names = ["surface area", "volume", "height", "floor area"]

obj[:,1] = -obj[:,1]
obj[:,2] = -obj[:,2]
obj[:,3] = -obj[:,3]
parallel(obj, axis_names)

#%%

import pandas as pd
df = pd.DataFrame(var)
df.to_csv("decision_vectors_4.csv")

df = pd.DataFrame(obj)
df.to_csv("decision_objectives_4.csv")

#%%

import matplotlib.pyplot as plt

fig = plt.figure(figsize=(15,6))
plt.scatter(obj[:,0],obj[:,1])
plt.xlabel("Surface area")
plt.ylabel("Volume")
```

Plotting the results …
example tent visualization across the Pareto front






```python
from modules.GeometryDesign.tent import Tent
box_points = np.array([
        [0,0,0], [1,0,0], [0,1,0], [1,1,0], # floor corners
        [0,0,1], [1,0,1], [0,1,1], [1,1,1] # Ceiling/roof corners
    ])
print(box_points.shape)

i=0
for coordenadas in var:
    print(obj[i,1])
    box_points = np.reshape(coordenadas, (15, 3))
    # Instansiate the object
    box = Tent(box_points)
    # Plot the box
    box.plot()
    i+=1
```

Take home message: Design optimization

- Understand the idea how optimization is used in design processes
- Discovery of optimal structures relates to the platonic world (ideal geometrical shapes, mathematical objects)
- Innovization (Innovation by Optimization) lets us learn about fundamental design principles
- Often simulators are coupled to optimizers in design optimization
- Black box objectives and constraints: Penalization of constraint violations ‘seen’ after simulation (implicit constraints)
- Parameterization to encode solutions and geometrical shapes by means of vectors: e.g. implicit formulae (Gielis’ superspheres), Bezier curves, convex hulls
- Visualization of results supports learning process about design space: example PC diagram of concatenated objective and decision space; visualization of typical designs in significant regions on the Pareto front.