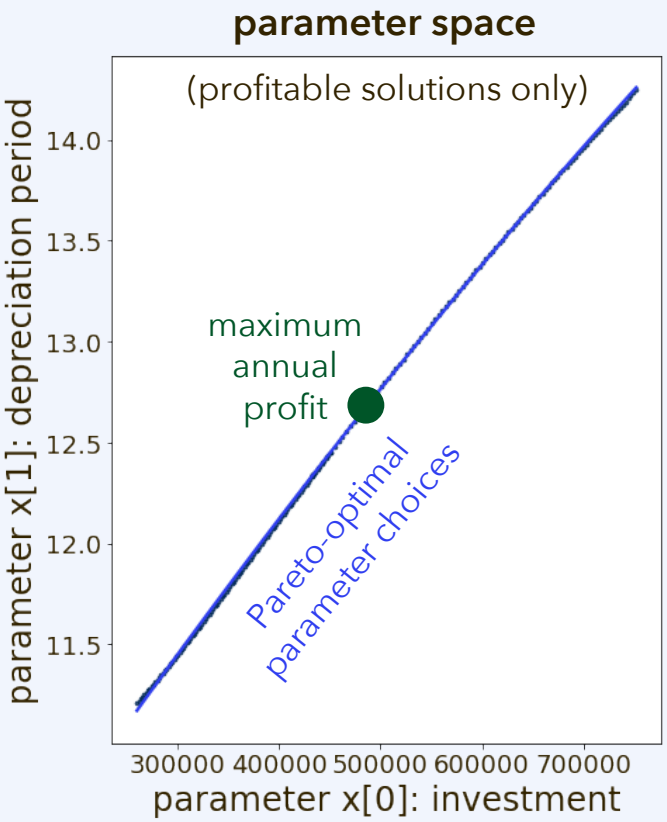
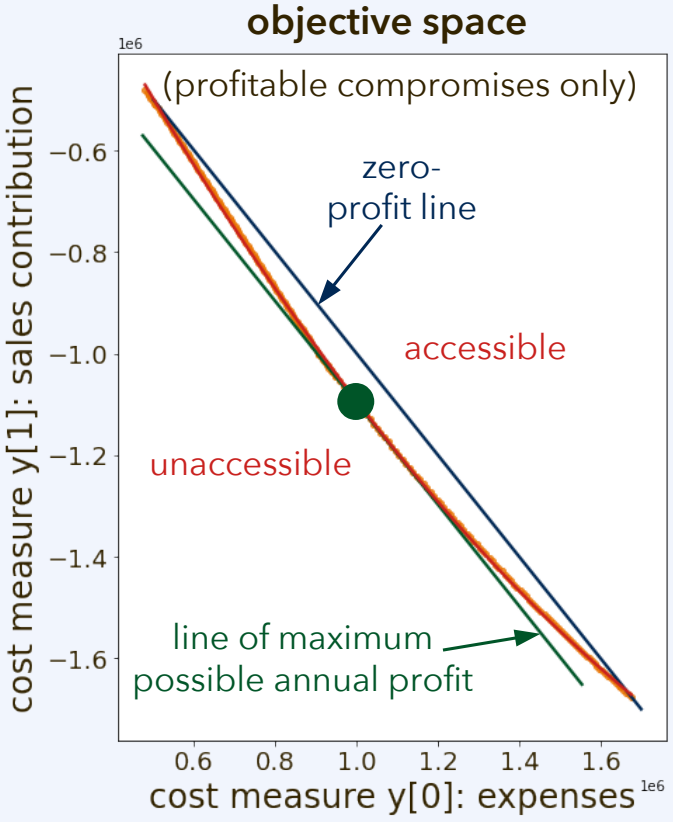
Pareto front visualization

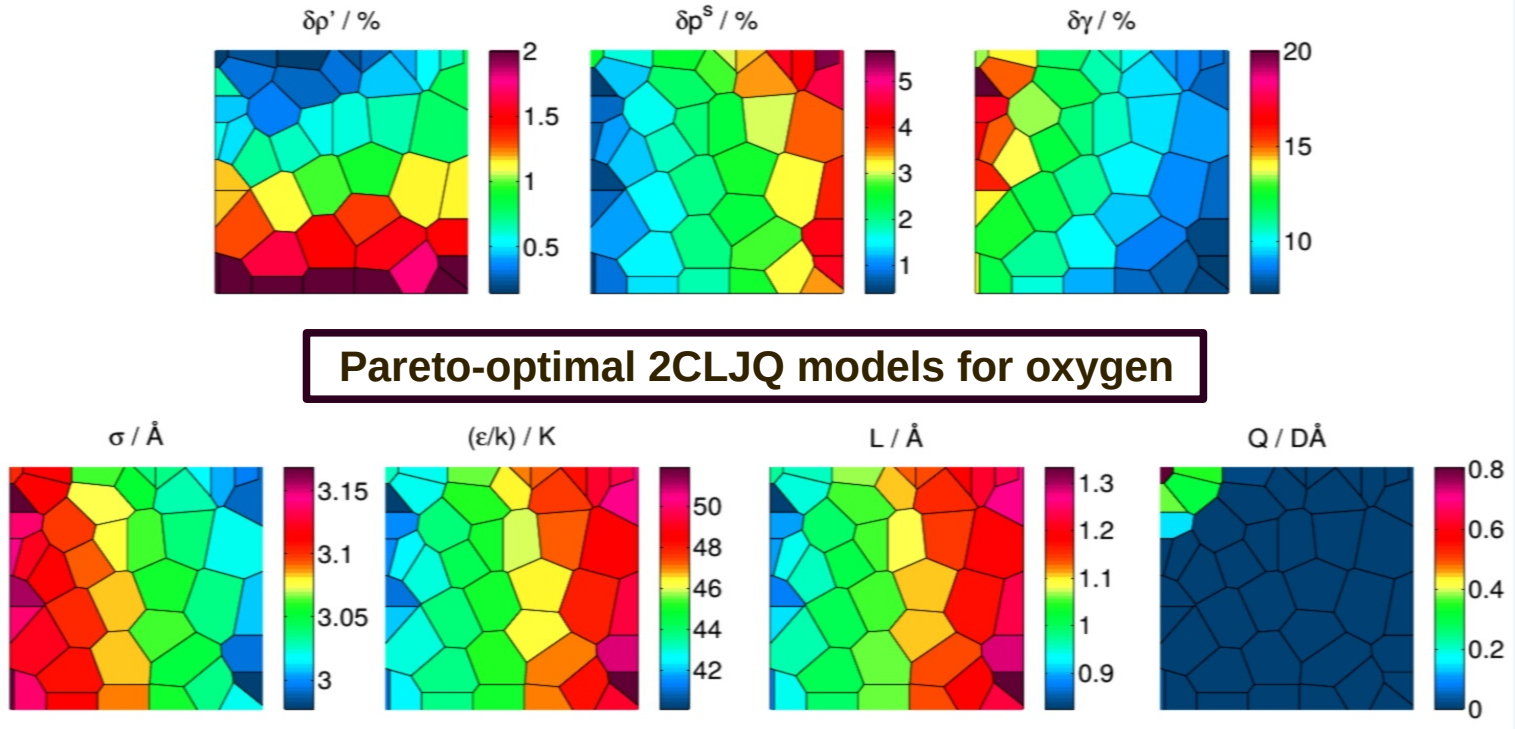
We have seen two visualization techniques, the first of which is applicable for 2D parameter and objective spaces only – it may be extended to 3D if an appropriate representation is used, *e.g.*, one that permits rotating the spaces.





Pareto front visualization

We have seen two visualization techniques. Using the second technique, each parameter and each objective is shown in its own heat map. Only the Pareto optimal solutions are shown; each solution corresponds to a field on the map.



objective space

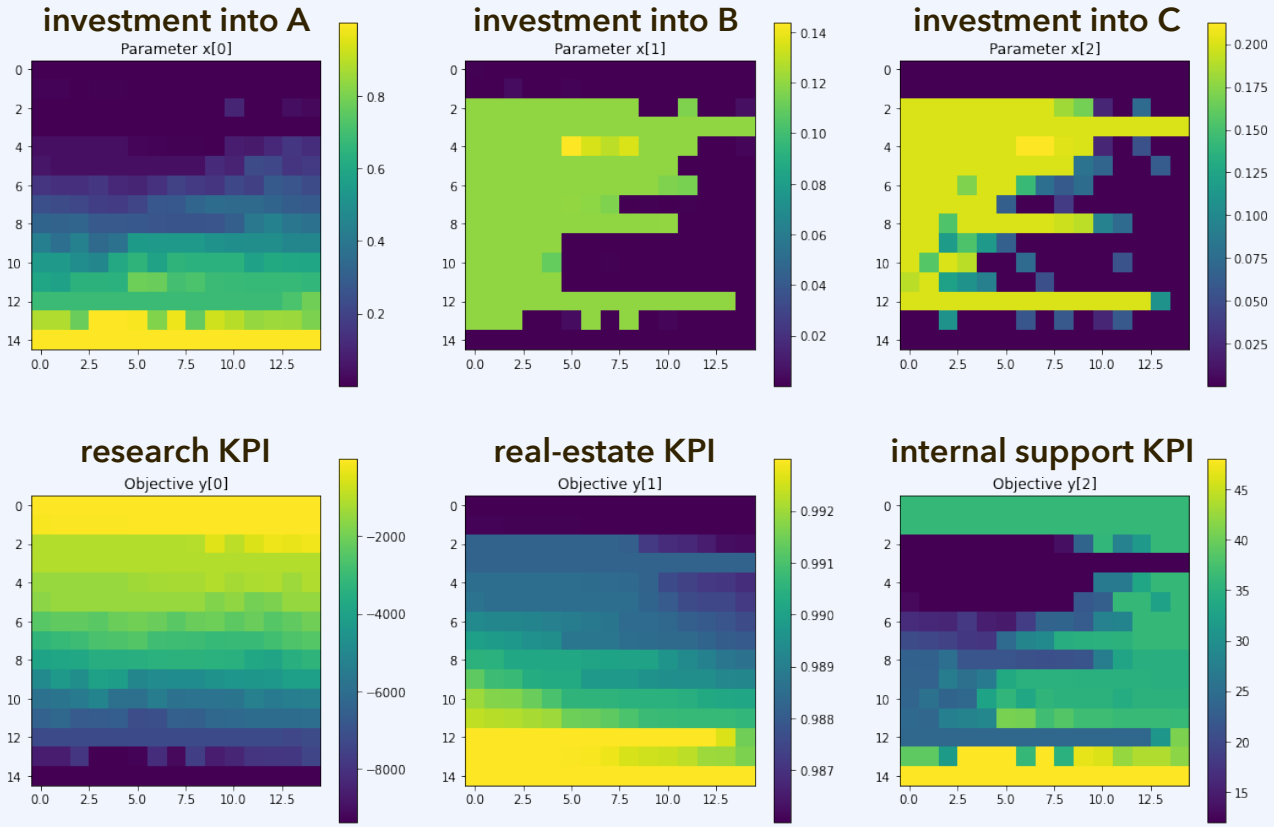
parameter space

K. Stöbener, P. Klein, M. Horsch, K. Küfer, H. Hasse, *Fluid Phase Equilib.* 411, 33 – 42, **2016**.

University of Central Lancashire
UCLan

Pareto front visualization: How to use the notebook

We have seen two visualization techniques. Using the second technique, each parameter and each objective is shown in its own heat map. Only the Pareto optimal solutions are shown; each solution corresponds to a field on the map.



University of Central Lancashire
UCLan

Example modelling problem

We have seen two visualization techniques. Using the second technique, each parameter and each objective is shown in its own heat map. Only the Pareto optimal solutions are shown; each solution corresponds to a field on the map.

The code from the **pareto-front-visualization** notebook is an ad-hoc creation. We will now go through the changes that need to be made when using it.

Example modelling scenario (based on Katib Hussain's case):

"The university has considered to upgrade workstations over the Christmas holidays (2 weeks). Up to 600 workstations would need to be upgraded, but for every ten workstations, a day's maintenance is required."

"A large maintenance team with too little investment would reduce productivity and reduce the amount of workstation upgrades. Too much investment with a small team would limit the use that can be made of the equipment."

University of Central Lancashire
UCLan

How to use the p.v. notebook (T2.2 discussion)

```
def cost_function(x, debug_output):
    if x[0] < 0 or x[1] < 1 or x[0] < salary*x[1]:
        return [math.inf, math.inf]
    expenses = x[0]
    acquired_equipment = (x[0] - salary*x[1]) / unit_cost
    upgraded_units = min(num_units, acquired_equipment, x[1]/fte_per_unit)

    y = [expenses, num_units - upgraded_units]
    return y
```

- In cell [1], replace the body of cost_function(**x**, debug_output).
- The constant coefficients need to be included.
- It is advisable to implement a **penalty for values outside the specified parameter space**, since scipy.optimize will not be aware of constraints.

University of Central Lancashire
UCLan

How to use the p.v. notebook (T2.2 discussion)

```
def random_parameters():
    max_expenses = num_units * (unit_cost + salary*fte_per_unit)
    expenses = random.uniform(0, max_expenses)
    total_labour_cost = random.uniform(0, expenses)
    return [expenses, total_labour_cost/salary]

objective_scale = [180000, 600]
sigma = 2
```

- In cell [1], replace the body of cost_function(**x**, debug_output).
- In cell [2], edit random_parameters() such that it returns a random point in parameter space, and **objective_scale** such that objective_scale[i] is of the order of variations expected in the outcome for objective y[i]. Increase/decrease sigma if you want weights to vary more/less.
- In cells [4] and [6], adjust local and global optimizer settings.

University of Central Lancashire
UCLan

How to use the p.v. notebook (T2.2 discussion)

In cell [6], adjust:

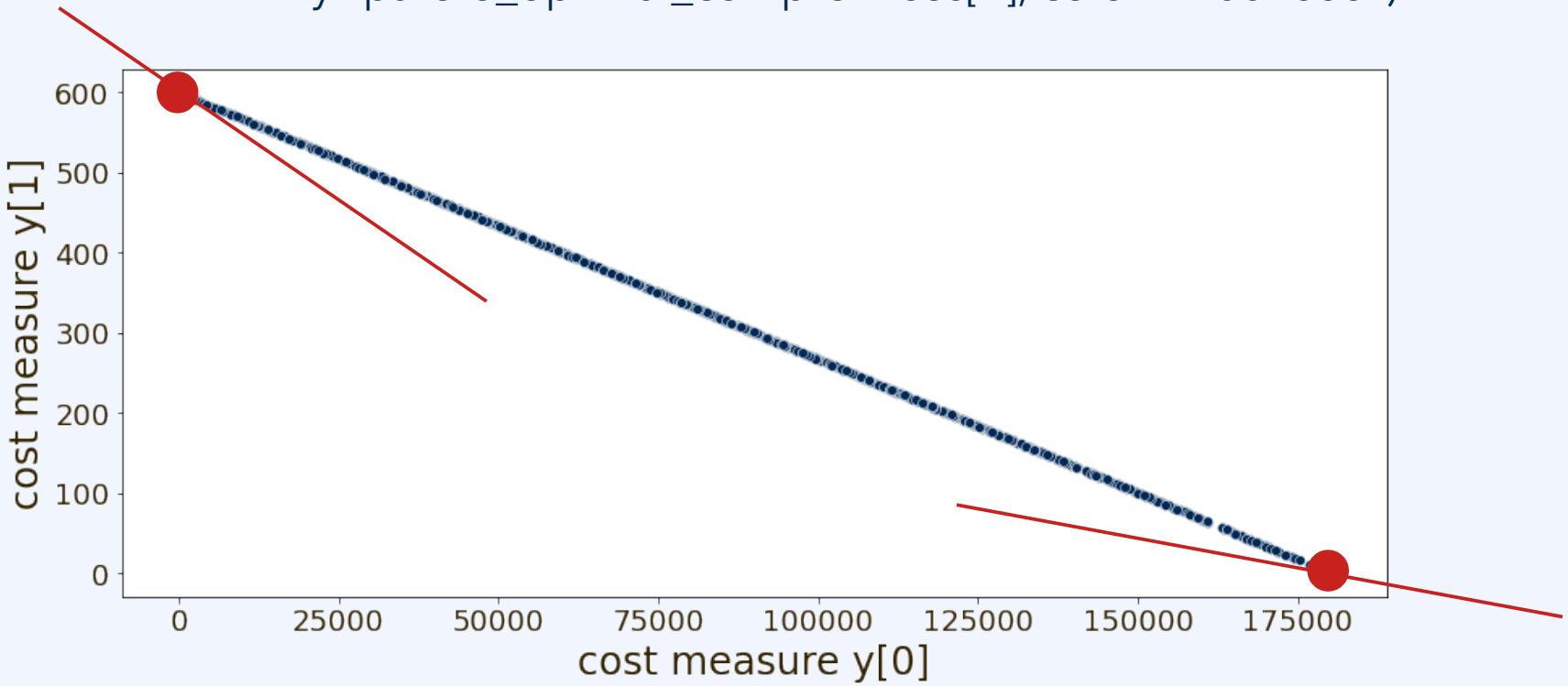
- number of parameters *m* and number of objectives *n*.
- number of points to be determined by linear combinations and by hyperboxing, respectively; their sum should be a square number.
    - linear combinations only work for a convex Pareto front: It can happen that this part needs to be removed; in this case, the lists **objective_space_lower** and **objective_space_upper** need to be initialized appropriately.
- local and global optimizer settings.

University of Central Lancashire
UCLan

How to use the p.v. notebook (T2.2 discussion)

In cell [8], select the axes to be shown for the 2D projection (here, 0 and 1).

```
sbn.scatterplot(x=pareto_optimal_compromises[0], \
                y=pareto_optimal_compromises[1], color="#002855")
```



How to use the p.v. notebook (T2.2 discussion)

In cell [10], set square_size to the square root of the number of determined Pareto optimal solutions. Pass indices of the criteria for ordering (here, 0 and 1):

idx_order = arrange_indices(square_size, n, pareto_optimal_compromises, 0, 1)