University of Central Lancashire
UCLan

CO3519
Artificial Intelligence

Pareto visualization FAQ
Linear regression
Regression analysis

Where opportunity creates success

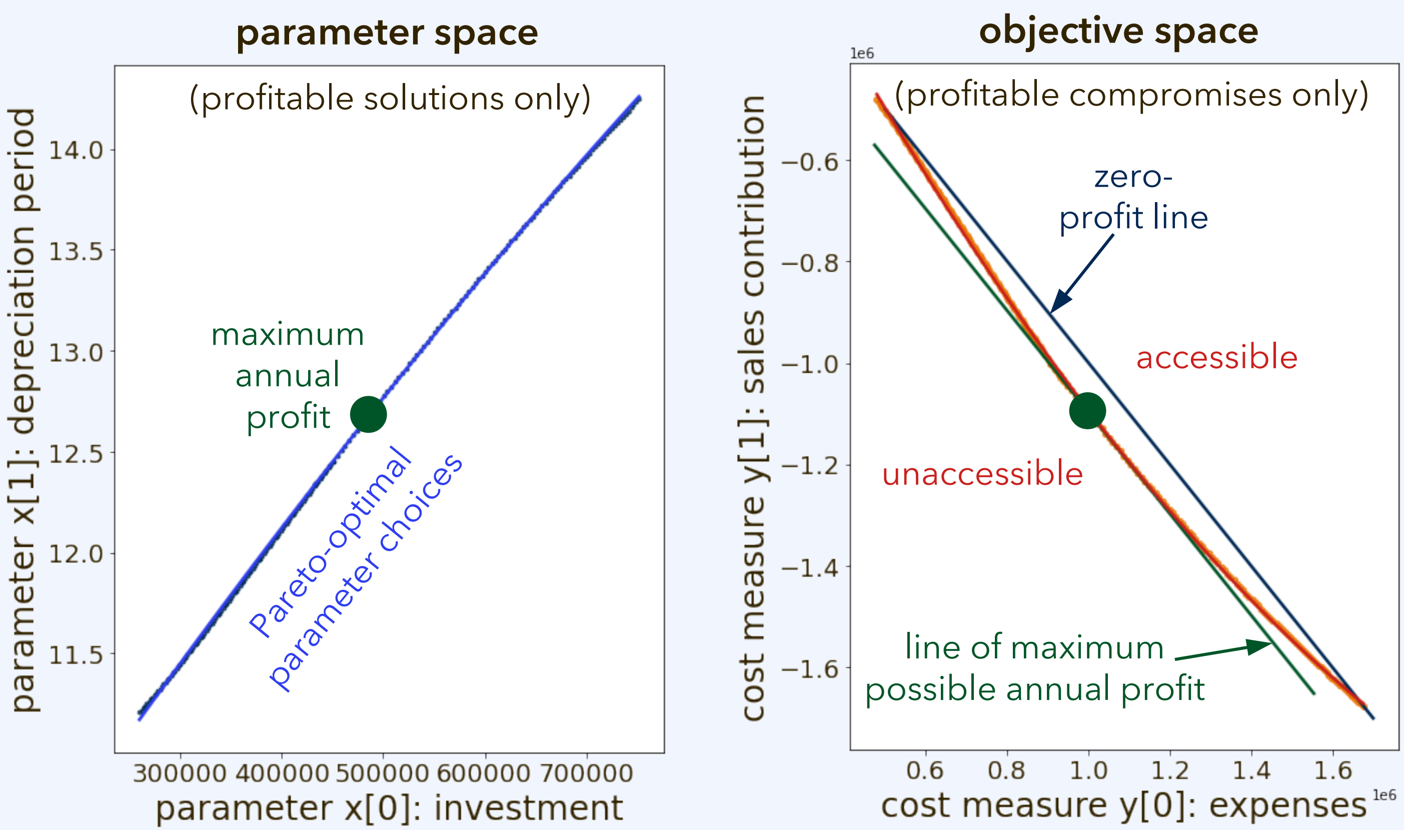
Pareto front visualization FAQ

CO3519

30th November 2021

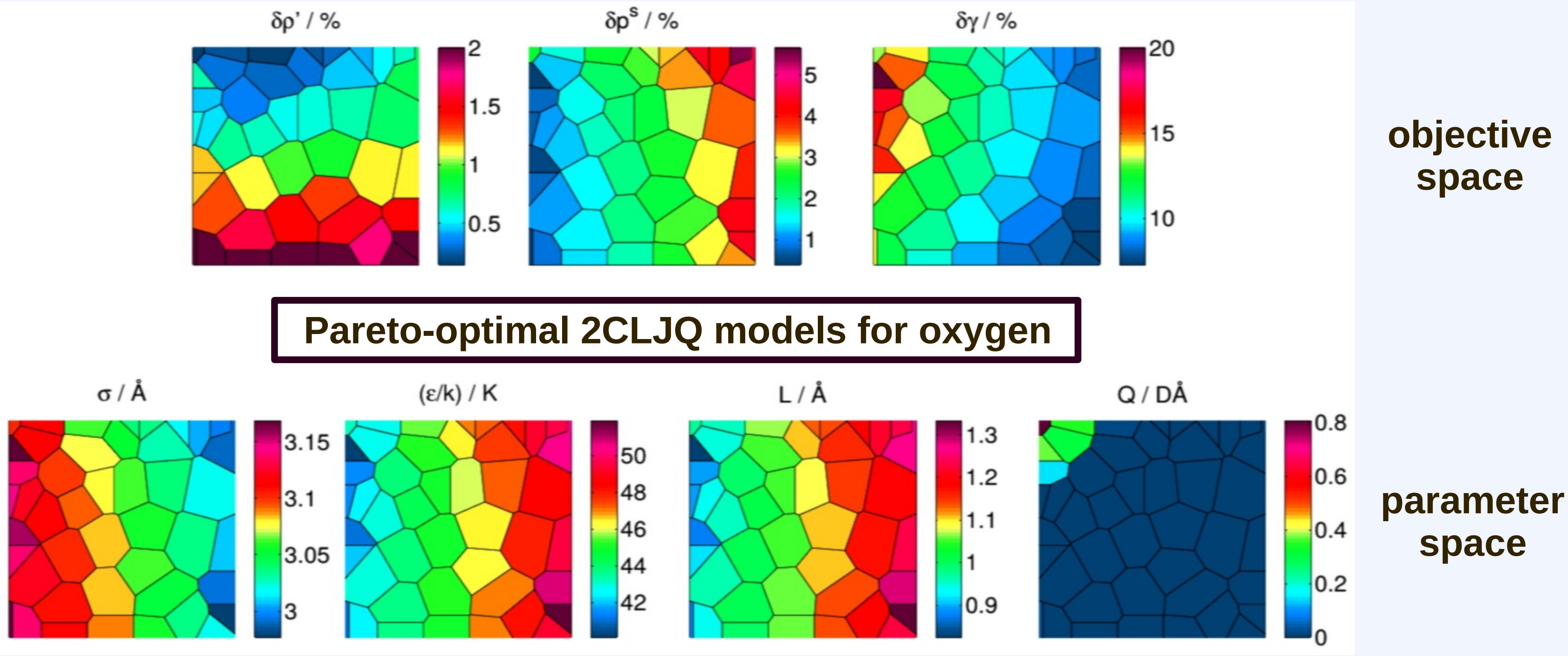
Pareto front visualization

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

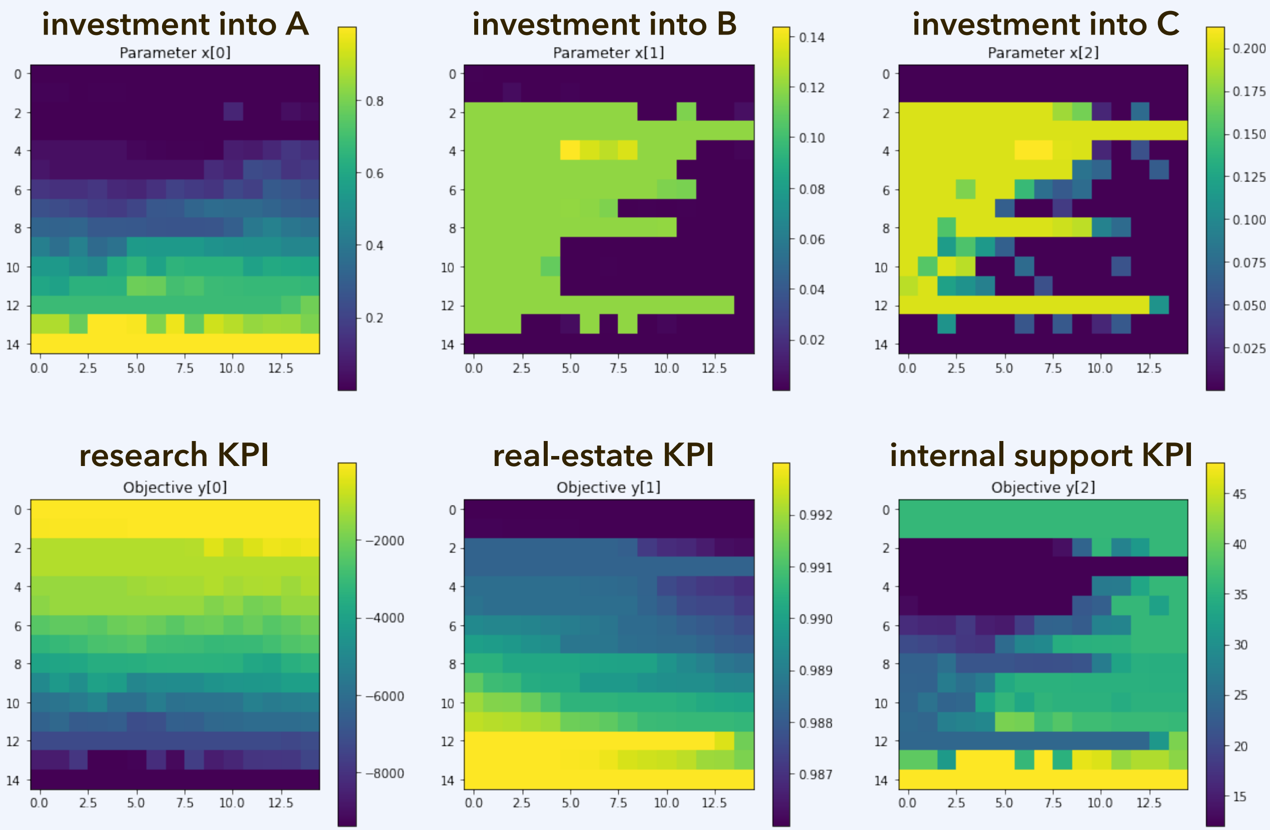
We have seen two visualization techniques. Using the second technique, each parameter and each objective is shown in its own heat map. Only the Pareto optimal solutions are shown; each solution corresponds to a field on the map.



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

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.



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."

Example modelling problem (T2.1 discussion)

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):

Minimization objectives:

- y_0, expenses of an upgrade/maintenance operation for workstations
- y_1, number of workstations (out of 600) that do not receive an upgrade

"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."

Example modelling problem (T2.1 discussion)

How to specify the parameter space: For each parameter, the range of permitted values needs to be specified; including constraints, if applicable.

Here, maybe simply $0 \leq x_0$ and $0 \leq x_1$.

Optional constraint, may be included explicitly or not: salary $\cdot\, x_1 \leq x_0$.

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

Minimization objectives:

- y_0, expenses of an upgrade/maintenance operation for workstations
- y_1, number of workstations (out of 600) that do not receive an upgrade

Parameters:

- x_0, expenses of an upgrade/maintenance operation for workstations
- x_1, number of staff assigned to carry out the upgrade (within two weeks)

Example modelling problem (T2.1 discussion)

What dimension do we expect for the relevant spaces and sets?

- The parameter space ($m = 2$) and objective space ($n = 2$) are both 2D.
- The accessible part of objective space ($q = 2$) will probably be 2D.
- Pareto front ($p = 1$) and set of Pareto-optimal solutions ($p' = 1$) are 1D.

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

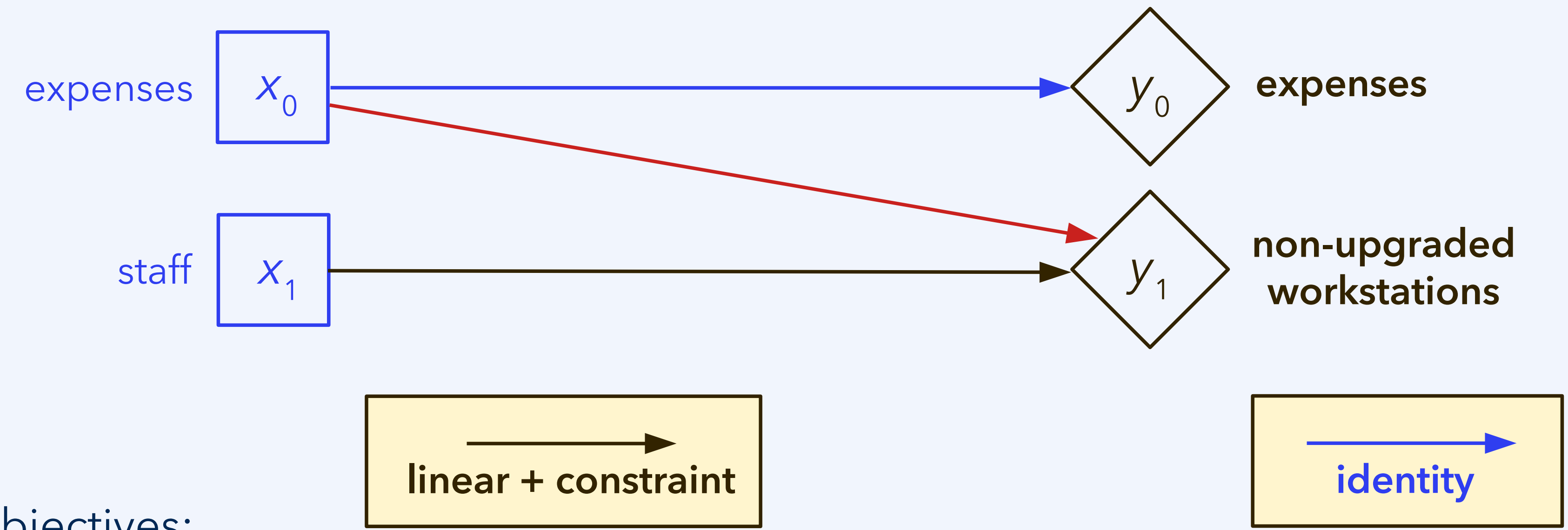
Minimization objectives:

- y_0, expenses of an upgrade/maintenance operation for workstations
- y_1, number of workstations (out of 600) that do not receive an upgrade

Parameters:

- x_0, expenses of an upgrade/maintenance operation for workstations
- x_1, number of staff assigned to carry out the upgrade (within two weeks)

Example modelling problem (T2.1 discussion)



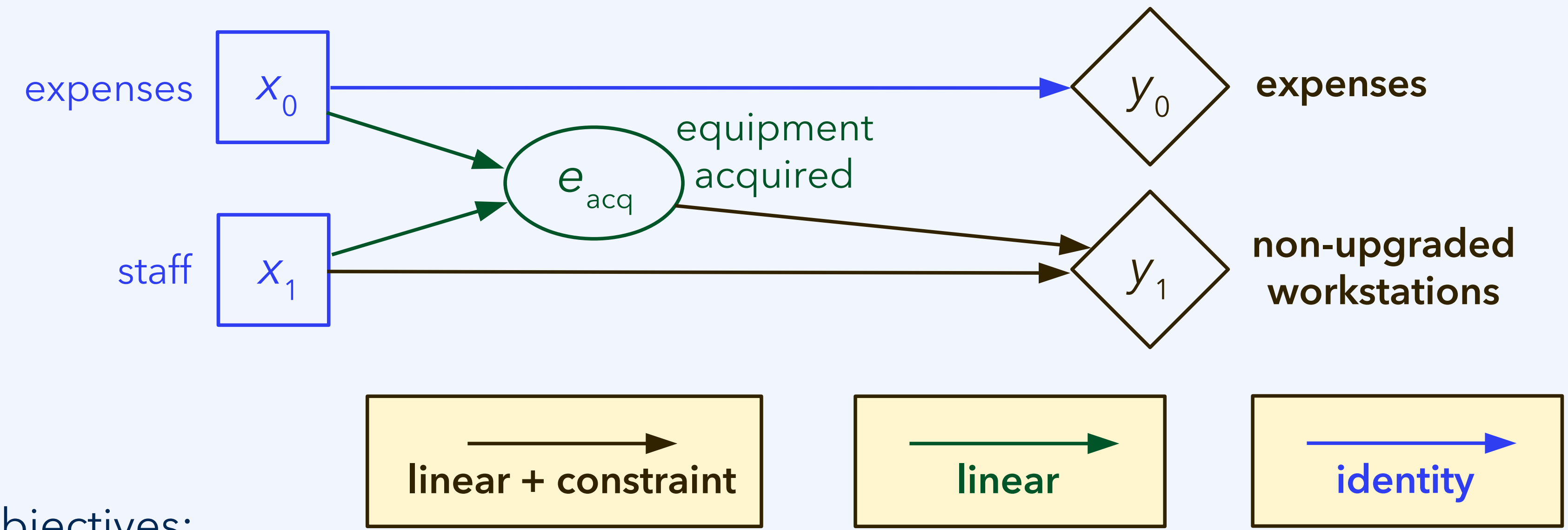
Objectives:

- y_0, expenses of an upgrade/maintenance operation for workstations
- y_1, number of workstations (out of 600) that do not receive an upgrade

Parameters:

- x_0, expenses of an upgrade/maintenance operation for workstations
- x_1, number of staff assigned to carry out the upgrade (within two weeks)

Example modelling problem (T2.1 discussion)



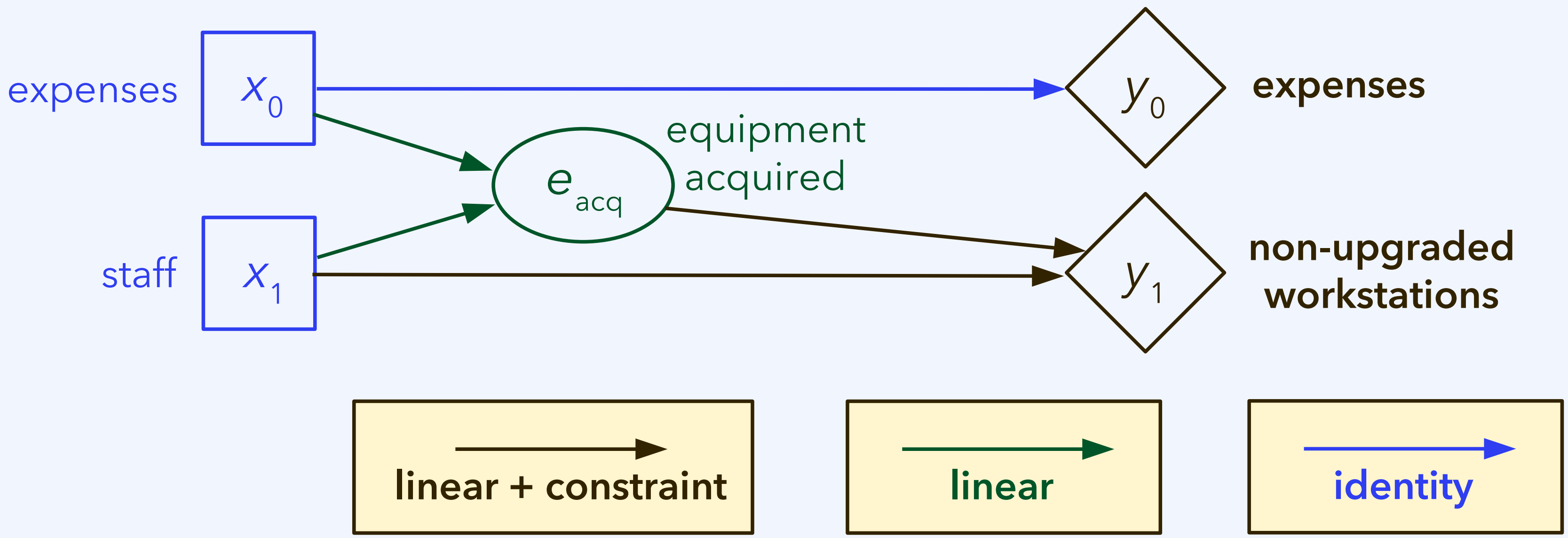
Objectives:

- y_0, expenses of an upgrade/maintenance operation for workstations
- y_1, number of workstations (out of 600) that do not receive an upgrade

Parameters:

- x_0, expenses of an upgrade/maintenance operation for workstations
- x_1, number of staff assigned to carry out the upgrade (within two weeks)

Example modelling problem (T2.1 discussion)



$y_0(x_0) = x_0$

$e_{acq}(x_0, x_1) = (x_0 - \text{salary} \cdot x_1) / \text{unit\_cost}$

$y_1(e_{acq}, x_1) = \max(0, \text{num\_units} - \min(e_{acq}, x_1/\text{fte\_per\_unit}))$

Example modelling problem (T2.1 discussion)

Reasonable estimates for the **constant coefficients** included in the model:

- num_units = 600, since 600 workstations require an upgrade.
- salary = GBP 15,000, *i.e.*, 1.0 FTE for two weeks for a technician. This is strictly speaking not just the salary, it needs to include non-wage costs.
- fte_per_unit = 0.01, since "for every ten workstations, a day's maintenance is required." Hence, 1.0 FTE can fix 100 units in ten working days.
- unit_cost = GBP 150, might be reasonable for a typical maintenance.

$$y_0(x_0) = x_0$$

$$e_{acq}(x_0, x_1) = (x_0 - \text{salary} \cdot x_1) \,/\, \text{unit\_cost}$$

$$y_1(e_{acq}, x_1) = \max(0, \text{num\_units} - \min(e_{acq}, x_1/\text{fte\_per\_unit}))$$

Example modelling problem (T2.1 discussion)

Can this scenario be treated by multicriteria optimization?
Verify whether the optimization criteria are in mutual disagreement:

- Optimum with respect to y_0 only, ignoring y_1:
  No expenses, therefore also no staff allocated to the work ($x_0 = x_1 = 0$).
- Optimum with respect to y_1 only, ignoring y_0: Any solution where all workstations are upgraded. There, $x_1 \geq \text{num\_units} \cdot \text{fte\_per\_unit} = 6$. Also, $x_0 \geq \text{num\_units} \cdot \text{unit\_cost} + x_1 \cdot \text{salary} \geq$ GBP 180,000.

$$y_0(x_0) = x_0$$

$$e_{acq}(x_0, x_1) = (x_0 - \text{salary} \cdot x_1) \,/\, \text{unit\_cost}$$

$$y_1(e_{acq}, x_1) = \max(0, \text{num\_units} - \min(e_{acq}, x_1/\text{fte\_per\_unit}))$$

Example modelling problem (T2.1 discussion)

```
def cost_function(x):

    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
```

$y_0(x_0) = x_0$

$e_{acq}(x_0, x_1) = (x_0 - \text{salary} \cdot x_1) \,/\, \text{unit\_cost}$

$y_1(e_{acq}, x_1) = \max(0, \text{num\_units} - \min(e_{acq}, x_1/\text{fte\_per\_unit}))$

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.

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

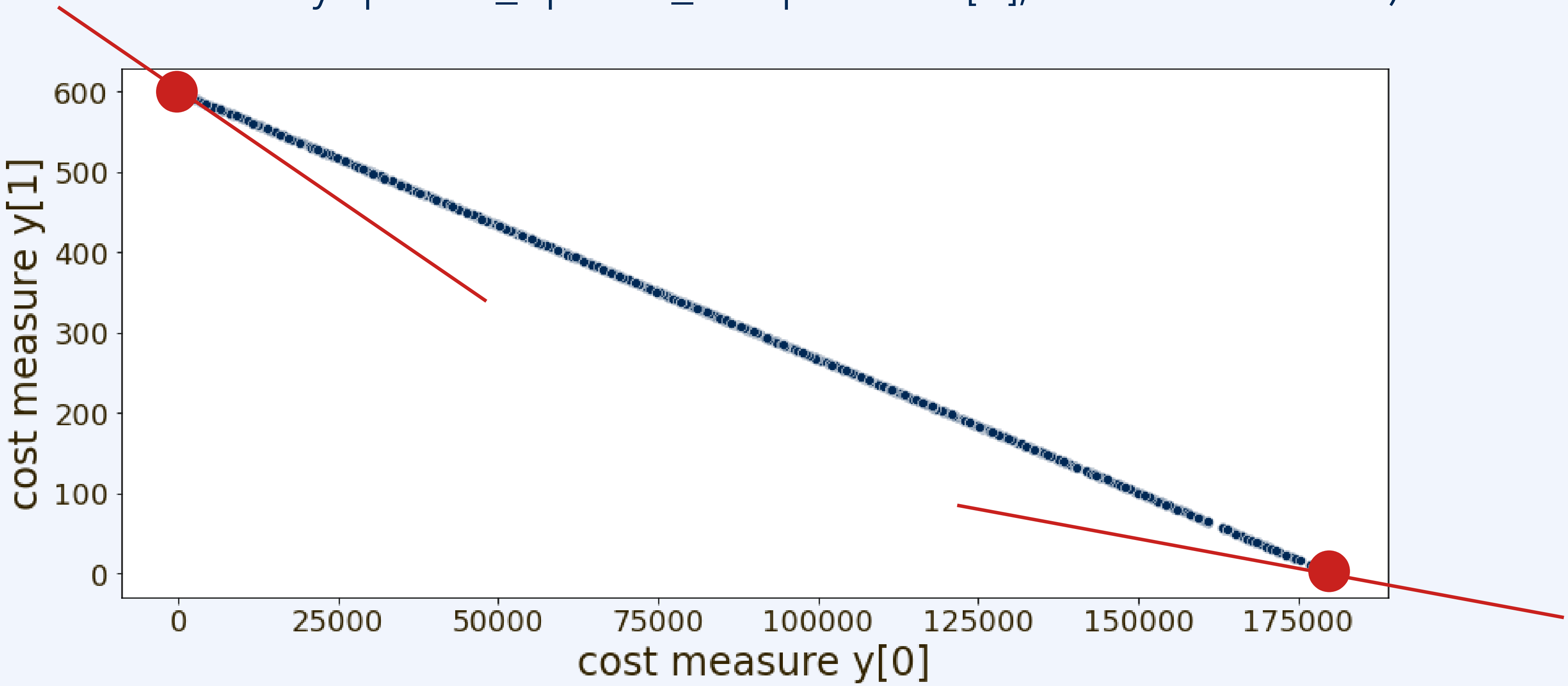
In cell [6], adjust:

- 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.

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).

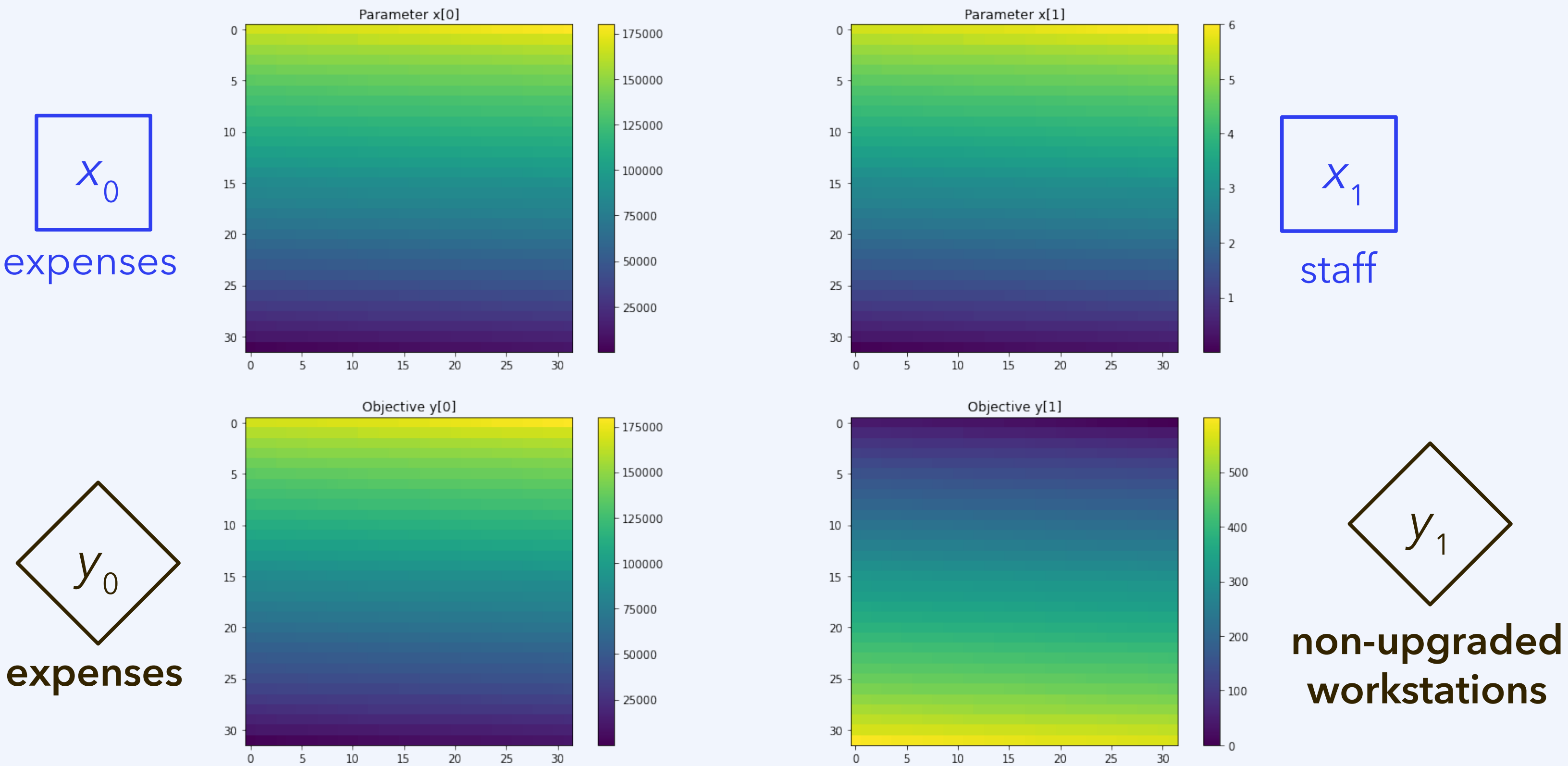
```
sbn.scatterplot(x=pareto_optimal_compromises[0], \
                y=pareto_optimal_compromises[1], color="#002855")
```



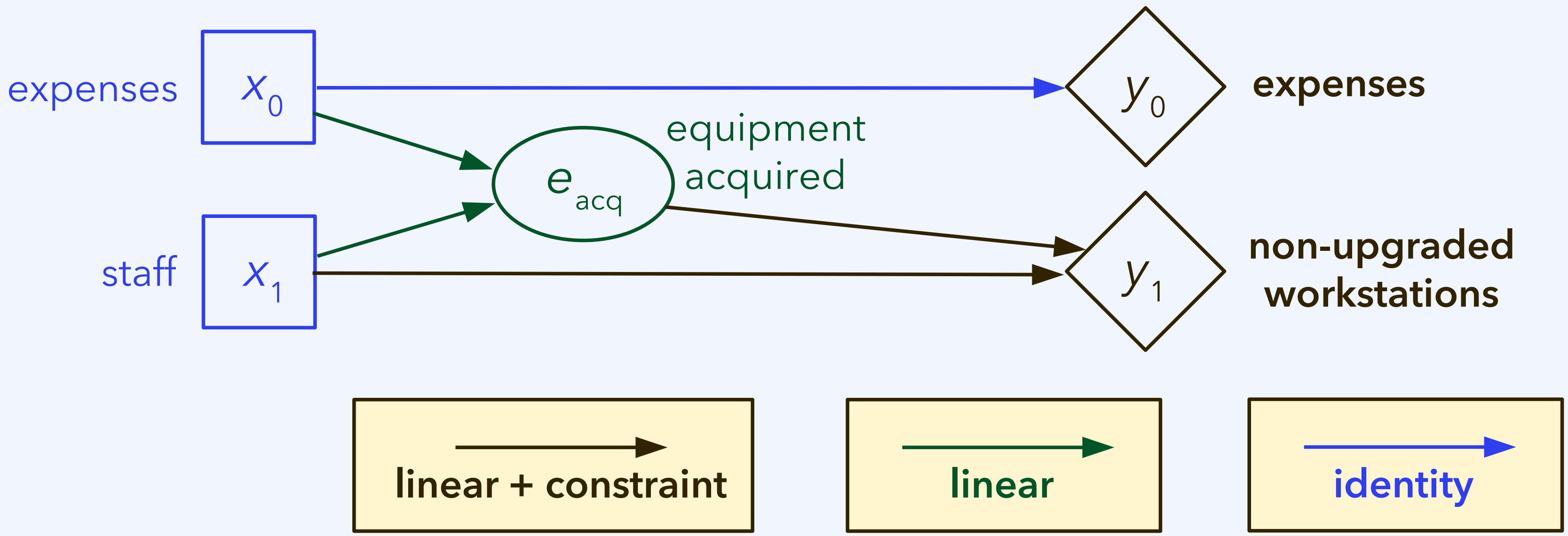
How to use the p.v. notebook (T2.2 discussion)

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)



Reanalysis of the problem (T2.1 discussion)

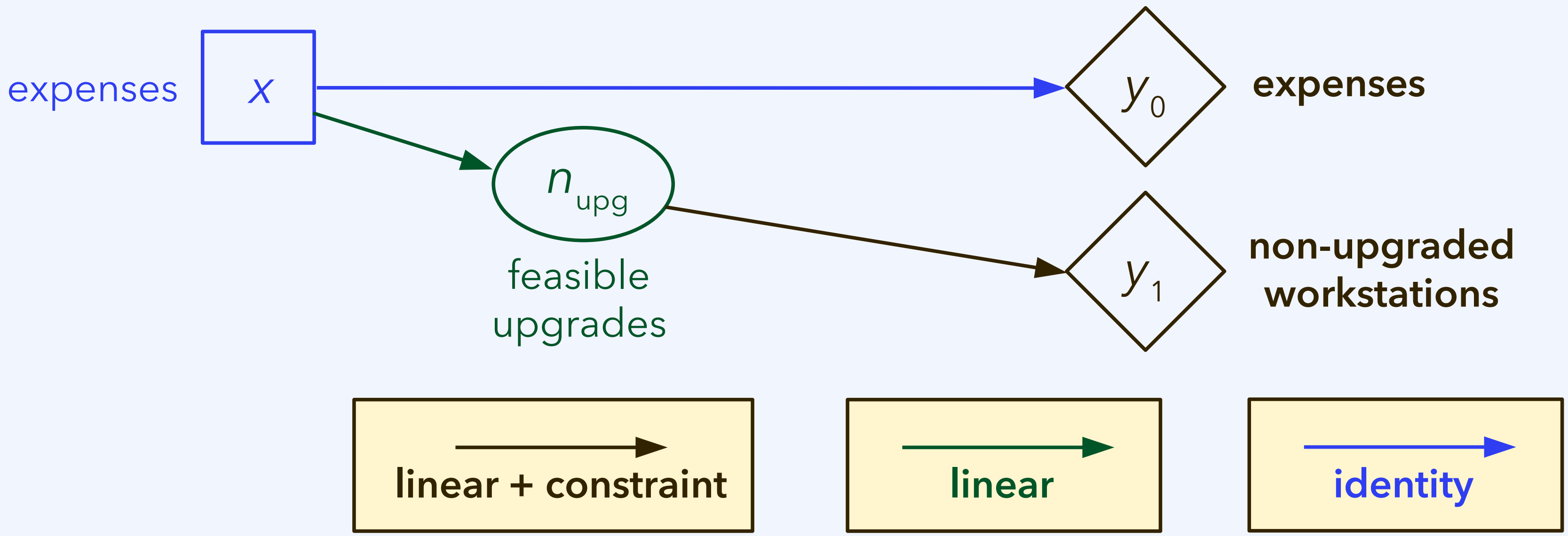


$y_0(x_0) = x_0$

$e_{acq}(x_0, x_1) = (x_0 - \text{salary} \cdot x_1) / \text{unit\_cost}$

$y_1(e_{acq}, x_1) = \max(0, \text{num\_units} - \min(e_{acq}, x_1/\text{fte\_per\_unit}))$

Reanalysis of the problem (T2.1 discussion)



$y_0(x) = x$

$n_{upg}(x) = x \,/\, (\text{unit\_cost} + \text{salary} \cdot \text{fte\_per\_unit})$

$y_1(n_{upg}) = \max(0, \text{num\_units} - n_{upg})$

Linear regression

CO3519

30th November 2021

Learning from data

Common aims in modelling are for a model (e.g., an objective function) to be

- **quantitatively** accurate, both for
  - descriptions, *i.e.*, it should reproduce the known data correctly,
  - predictions, *e.g.*, for interpolation and extrapolation from data.
- **qualitatively** accurate, *i.e.*, it should correctly reflect *the way* in which multiple variables relate to each other.

These expectations very roughly relate to the two main modes of reasoning:

- **inductive** reasoning, where conclusions are drawn from patterns in data sets or statistics over data: This is what we here mean by "learning."
- **deductive** reasoning, also just "reasoning," where a premise (logically, mathematically) implies the conclusion, which is thus rigorously proven.

Learning from data

Common aims in modelling are for a model (e.g., an objective function) to be

- **quantitatively** accurate, both for
  - descriptions, *i.e.*, it should reproduce the known data correctly,
  - predictions, *e.g.*, for interpolation and extrapolation from data.

Qualitative accuracy relies on theories, quantitative accuracy on empirical data.

These expectations very roughly relate to the two main modes of reasoning:

- **inductive** reasoning, where conclusions are drawn from patterns in data sets or statistics over data: This is what we here mean by "learning."

Deductive reasoning relies on theories, learning relies on empirical data.

Learning from data

Categorization of learning methods:

– **Supervised learning**, where an agent obtains input-output pairs directly or indirectly from its percepts; *e.g.*, lists **x** and **y** are taken from sensory input, and a model $f(\mathbf{x}) = \mathbf{y}_{\text{model}}$ is constructed, aiming toward $\mathbf{y}_{\text{model}} = \mathbf{y}$. The model function is not arbitrary, but based on a priori **hypotheses**.

The **model quality** can be assessed by **validation and testing**, *i.e.*, by evaluating how well the model predicts data on which it has not been trained.

Learning from data

Categorization of learning methods:

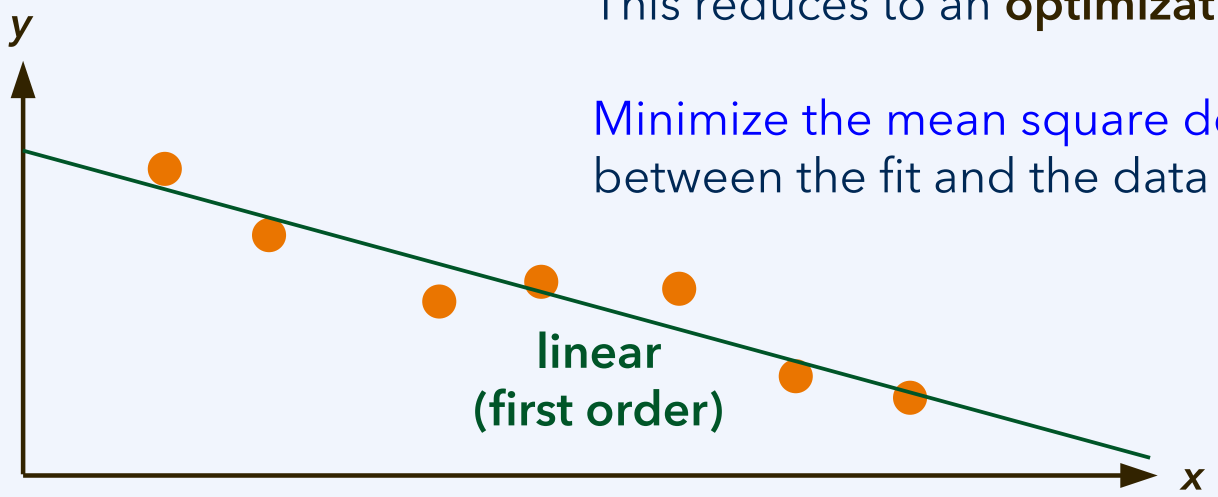
- **Supervised learning**, where an agent obtains input-output pairs directly or indirectly from its percepts; *e.g.*, lists $\mathbf{x}$ and $\mathbf{y}$ are taken from sensory input, and a model $f(\mathbf{x}) = \mathbf{y}_{\text{model}}$ is constructed, aiming toward $\mathbf{y}_{\text{model}} = \mathbf{y}$. The model function is not arbitrary, but based on a priori **hypotheses**.
- **Unsupervised learning**, where lists of variable values $\mathbf{x}_0, \ldots, \mathbf{x}_n$ are given to the agent/algorithm without any a priori hypotheses. It is up to the agent/algorithm to detect any patterns in the data set autonomously.
- **Reinforcement learning**, like the above, but with feedback on the model quality provided to the agent at each iteration.

It is possible to combine these approaches, *e.g.*, by providing some a priori hypotheses about how the world functions, but not enough for a complete model.

Learning from data by regression

Data are typically affected by noise, random error, fluctuations, and similar phenomena that obscure to what extent variables are related to each other.

Regression analysis can help recover the **correlations between variables**.



This reduces to an **optimization problem**:

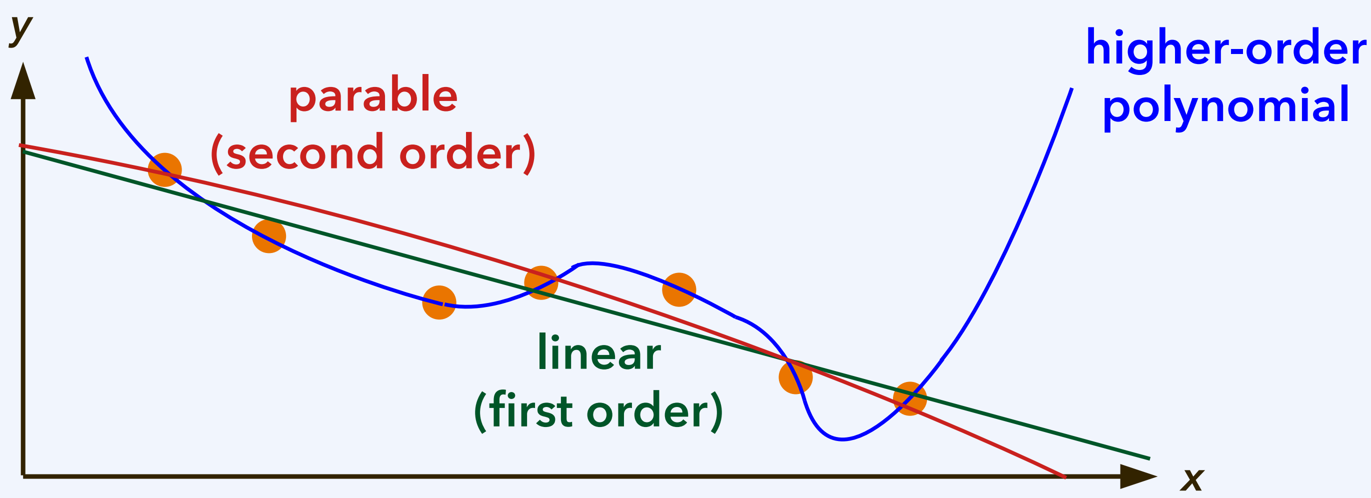
Minimize the mean square deviation between the fit and the data points.

This is also called an **ordinary least squares (OLS)** fit of a line to a data set.

Learning from data by regression

Data are typically affected by noise, random error, fluctuations, and similar phenomena that obscure to what extent variables are related to each other.

Regression analysis can help recover the **correlations between variables**.



In supervised learning, the user specifies the type of model (*i.e.*, the **hypothesis**).

Learning from data by linear regression

Data are typically affected by noise, random error, fluctuations, and similar phenomena that obscure to what extent variables are related to each other.

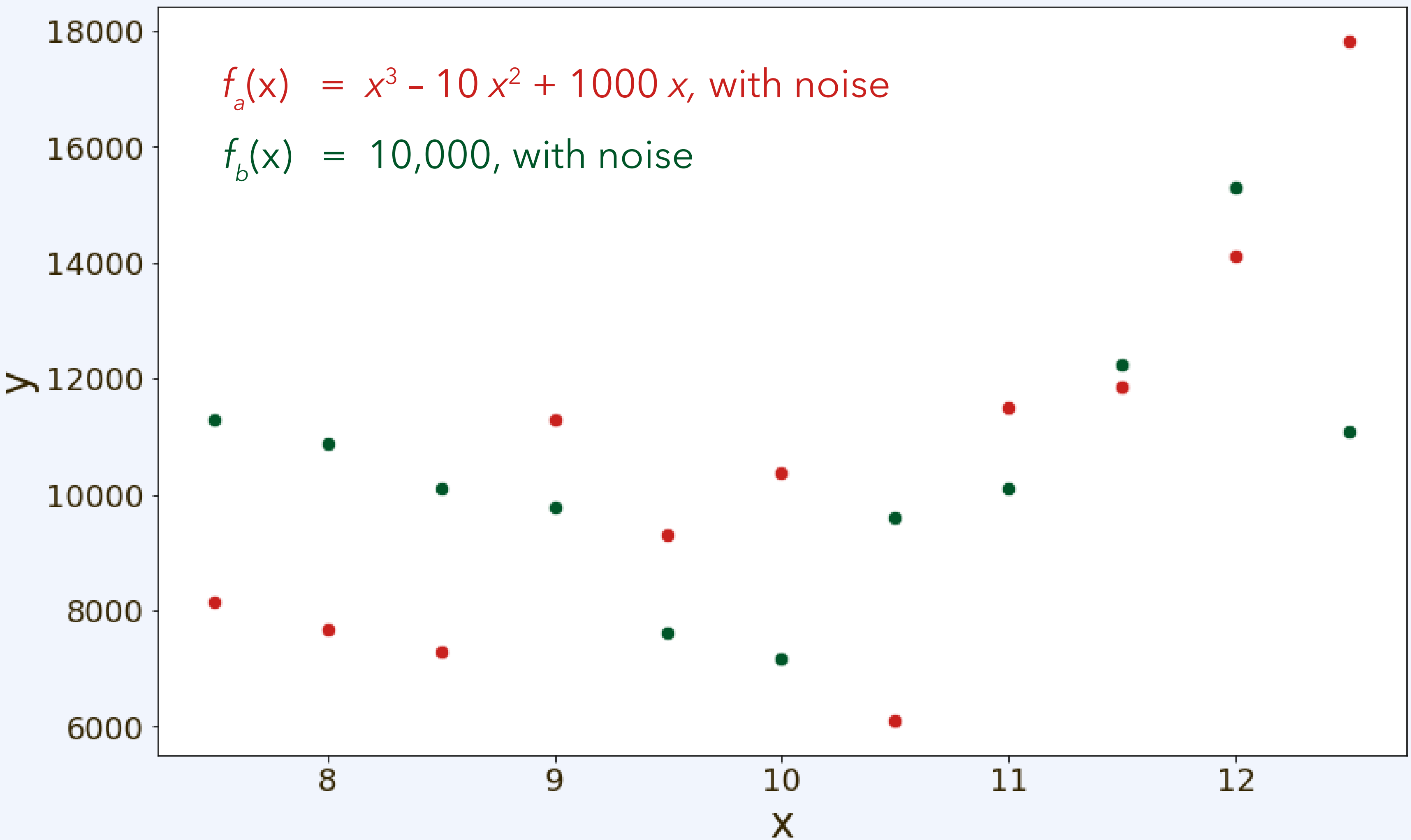
Regression analysis can help recover the **correlations between variables**, and it can state how probable it is that such an underlying relationship is actually present, as opposed to just being noise. It can be used in any form of learning, but is most effective as a **supervised learning** method.

The most straightforward, but nonetheless very powerful method for this purpose is **linear regression**. As an example, we consider two data sets, each generated by one of the following functions and affected by substantial noise:

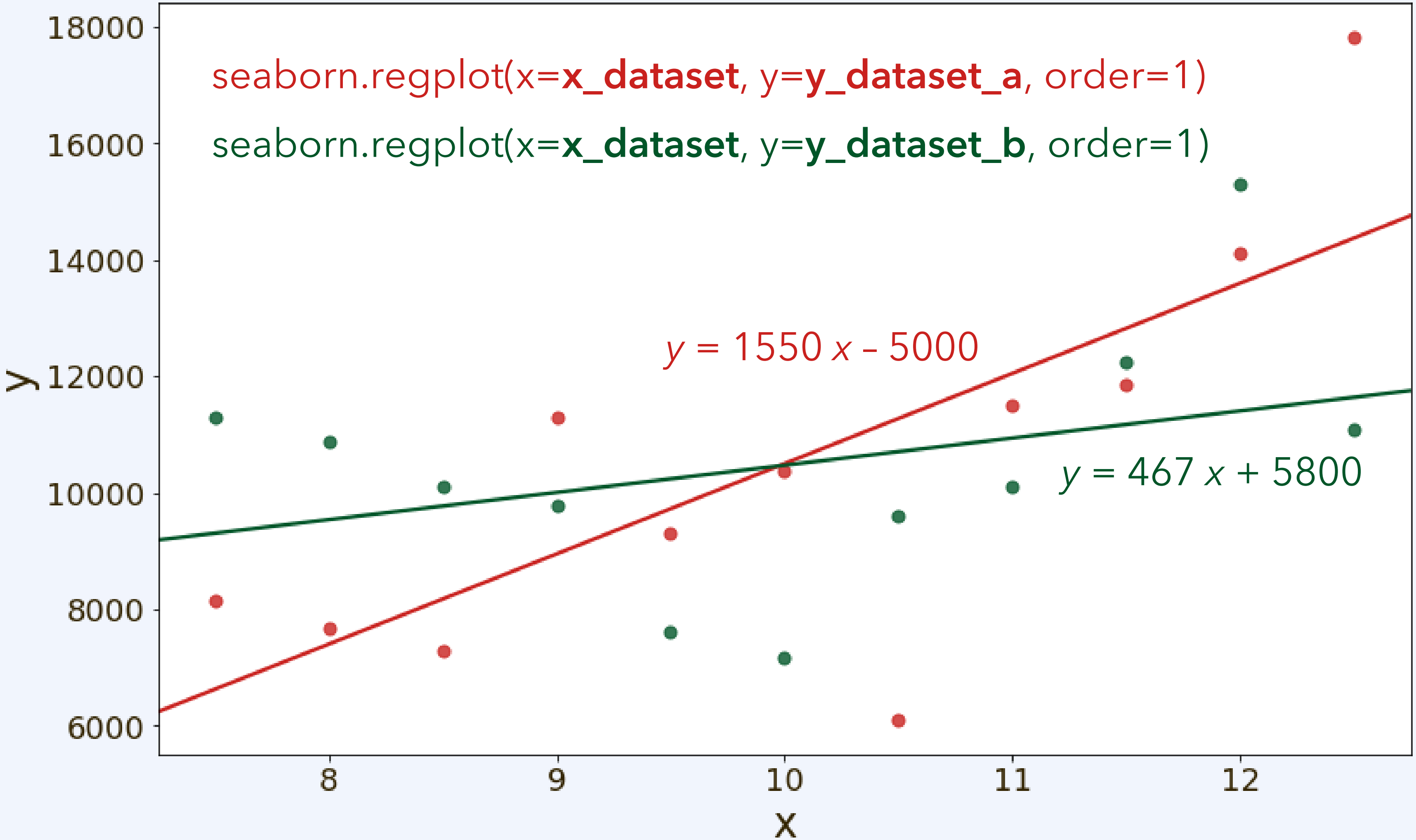
$$f_a(x) = x^3 - 10\,x^2 + 1000\,x$$

$$f_b(x) = 10{,}000$$

Learning from data by linear regression



Learning from data by linear regression



Regression analysis

CO3519

30th November 2021

Is there a correlation or is it only noise?

```
import statsmodels.api as sm

x_array = sm.add_constant(np.asarray(x_dataset))
linear_fit_a = sm.OLS(np.asarray(y_dataset_a), x_array).fit()

print("Fit a):\n", linear_fit_a.summary())
```

$y = 1550\,x - 5000$

```
Fit a):
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.574
Model:                            OLS   Adj. R-squared:                  0.526
Method:                 Least Squares   F-statistic:                     12.11
Date:                Mon, 29 Nov 2021   Prob (F-statistic):            0.00693
Time:                        15:11:38   Log-Likelihood:                -99.816
No. Observations:                  11   AIC:                             203.6
Df Residuals:                       9   BIC:                             204.4
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -4997.9028   4507.307     -1.109      0.296   -1.52e+04    5198.333
x1          1549.5537    445.200      3.481      0.007     542.441    2556.666
==============================================================================
Omnibus:                        5.158   Durbin-Watson:                   1.606
Prob(Omnibus):                  0.076   Jarque-Bera (JB):                1.722
Skew:                          -0.797   Prob(JB):                        0.423
Kurtosis:                       4.103   Cond. No.                         65.4
==============================================================================
```

Is there a correlation or is it only noise?

```python
import statsmodels.api as sm

x_array = sm.add_constant(np.asarray(x_dataset))
linear_fit_a = sm.OLS(np.asarray(y_dataset_a), x_array).fit()

print("Fit a):\n", linear_fit_a.summary())
```

```
Fit a):
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.574
Model:                            OLS   Adj. R-squared:                  0.526
Method:                 Least Squares   F-statistic:                     12.11
Date:                Mon, 29 Nov 2021   Prob (F-statistic):            0.00693
Time:                        15:11:38   Log-Likelihood:                -99.816
No. Observations:                  11   AIC:                             203.6
Df Residuals:                       9   BIC:                             204.4
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -4997.9028   4507.307     -1.109      0.296   -1.52e+04    5198.333
x1          1549.5537    445.200      3.481      0.007     542.441    2556.666
==============================================================================
Omnibus:                        5.158   Durbin-Watson:                   1.606
Prob(Omnibus):                  0.076   Jarque-Bera (JB):                1.722
Skew:                          -0.797   Prob(JB):                        0.423
Kurtosis:                       4.103   Cond. No.                         65.4
==============================================================================
```

$y = 1550\,x - 5000$

95% probability that the linear coefficient is between 542 and 2560

Is there a correlation or is it only noise?

University of Central Lancashire
UCLan

```python
import statsmodels.api as sm

x_array = sm.add_constant(np.asarray(x_dataset))
linear_fit_a = sm.OLS(np.asarray(y_dataset_a), x_array).fit()

print("Fit a):\n", linear_fit_a.summary())
```

```
Fit a):
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.574
Model:                            OLS   Adj. R-squared:                  0.526
Method:                 Least Squares   F-statistic:                     12.11
Date:                Mon, 29 Nov 2021   Prob (F-statistic):            0.00693
Time:                        15:11:38   Log-Likelihood:                -99.816
No. Observations:                  11   AIC:                             203.6
Df Residuals:                       9   BIC:                             204.4
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -4997.9028   4507.307     -1.109      0.296   -1.52e+04    5198.333
x1          1549.5537    445.200      3.481      0.007     542.441    2556.666
==============================================================================
Omnibus:                        5.158   Durbin-Watson:                   1.606
Prob(Omnibus):                  0.076   Jarque-Bera (JB):                1.722
Skew:                          -0.797   Prob(JB):                        0.423
Kurtosis:                       4.103   Cond. No.                         65.4
==============================================================================
```

$y = 1550\,x - 5000$

if the variables are independent, there is a **0.7%** probability of artificially creating (at least) such a strong correlation by chance

95% probability that the linear coefficient is between 542 and 2560

Is there a correlation or is it only noise?

University of Central Lancashire UCLan

Compare data set b), with no actual underlying correlation between x and y.

```
linear_fit_b = sm.OLS(np.asarray(y_dataset_b), x_array).fit()
print("Fit b):\n", linear_fit_b.summary())
```

```
Fit b):
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.124
Model:                            OLS   Adj. R-squared:                  0.026
Method:                 Least Squares   F-statistic:                     1.270
Date:                Mon, 29 Nov 2021   Prob (F-statistic):              0.289
Time:                        15:10:39   Log-Likelihood:                -99.022
No. Observations:                  11   AIC:                             202.0
Df Residuals:                       9   BIC:                             202.8
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       5801.8258   4193.444      1.384      0.200   -3684.404    1.53e+04
x1           466.8272    414.199      1.127      0.289    -470.156    1403.810
==============================================================================
Omnibus:                        0.210   Durbin-Watson:                   1.045
Prob(Omnibus):                  0.900   Jarque-Bera (JB):                0.143
Skew:                           0.182   Prob(JB):                        0.931
Kurtosis:                       2.577   Cond. No.                         65.4
==============================================================================
```

$y = 467\,x + 5800$

if the variables are independent, there is a **28.9%** probability of artificially creating (at least) such a strong correlation by chance

95% probability that the linear coefficient is between –470 and +1400

Is there a correlation or is it only noise?

Compare data set b), with no actual underlying correlation between x and y.

This quantity is called "the *p* value."

It indicates the probability of the same or a stonger apparent correlation between two variables (here, *x* and *y*), assuming that the null hypothesis is true.

Null hypothesis: There is no actual underlying correlation between x and y. Any appearance of such a correlation is due to chance.

if the variables are independent, there is a **28.9%** probability of artificially creating (at least) such a strong correlation by chance

95% probability that the linear coefficient is between –470 and +1400

Is there a correlation or is it only noise?

Compare data set b), with no actual underlying correlation between x and y.

This quantity is called "the *p* value."

It indicates the probability of the same or a stonger apparent correlation between two variables (here, *x* and *y*), assuming that the null hypothesis is true.

Null hypothesis: There is no actual underlying correlation between x and y. Any appearance of such a correlation is due to chance.

By convention, correlations are typically seen as statistically insignificant if $p > 5\%$.

if the variables are independent, there is a **28.9%** probability of artificially creating (at least) such a strong correlation by chance

95% probability that the linear coefficient is between –470 and +1400

Is there a correlation or is it only noise?

There are many potential **statistical fallacies** or traps inherent in this issue.

Assume we are particularly rigorous and require the p value to be lower than a level of significance of 0.01. That is, we only accept results that have a probability of 1% or less to have emerged by chance, given no actual correlation.

Now we instruct our high-throughput data analysis system to evaluate:

- Is there a correlation between avocado consumption and cancer? No.
- … between liver disease and number of pets in the household? No.

(… about a hundred more questions …)

Is there a correlation or is it only noise?

Now we instruct our high-throughput data analysis system to evaluate:

- Is there a correlation between avocado consumption and cancer? No.
- … between liver disease and number of pets in the household? No.
  (… about a hundred more questions …)
- … between coronary disease and consumption of elk meat? Yes, $p < 0.01$.

Next month in an illustrated paper: Eat elk meat to avoid heart attacks!
A scientific study has proven …

University of Central Lancashire
UCLan

CO3519
Artificial Intelligence

Pareto visualization FAQ
Linear regression
Regression analysis

Where opportunity creates success