Estimating intrinsic causal influences in real-world examples

This notebook demonstrates the usage of the intrinsic causal influence (ICC) method, a way to estimate causal influence in a system. A common question in many applications is: "What is the causal influence of node X on node Y?" Here, "causal influence" can be defined in various ways. One approach could be to measure the interventional influence, which asks, "How much does node Y change if I intervene on node X?" or, from a more feature relevance perspective, "How relevant is X in describing Y?"

In the following we focus on a particular type of causal influence, which is based on decomposing the generating process into mechanisms in place at each node, formalized by the respective causal mechanism. Then, ICC quantifies for each node the amount of uncertainty of the target that can be traced back to the respective mechanism. Hence, nodes that are deterministically computed from their parents obtain zero contribution. This concept may initially seem complex, but it is based on a simple idea:

Consider a chain of nodes: X -> Y -> Z. Y is more informative about Z than X, as Y directly determines Z and also incorporates all information from X. It is obvious that when intervening on either X or Y, Y has a more significant impact on Z. But, what if Y is just a rescaled copy of X, i.e., $Y = a \cdot X$? In this case, Y still has the largest interventional influence on Z, but it is not adding any new information on top of X. The ICC method, on the other hand, would attribute 0 influence to Y as it only passes on what it inherits from X.

The idea behind ICC is not to estimate the contribution of observed upstream nodes to the target node, but instead to attribute the influence of their noise terms. Since we model each node as a functional causal model of the form $X_i = f_i(PA_i, N_i)$, we aim to estimate the contribution of the N_i terms to the target. In the previous example, we have deterministic relationships with zero noise, i.e., the intrinsic influence is 0. This type of attribution is only possible when we explicitly model our causal relationships using functional causal models, as we do in the GCM module.

In the following, we will look at two real-world examples where we apply ICC.

Intrinsic influence on car MPG consumption

In the first example, we use the famous MPG data set, which contains different features that are used for the prediction of miles per gallon (mpg) of a car engine. The relationship between these features can be modeled as a graphical causal model. For this, we follow the causal graph defined in the work by Wang et al. and remove all nodes that have no influence on MPG. This leaves us with the following graph:

In [1]:

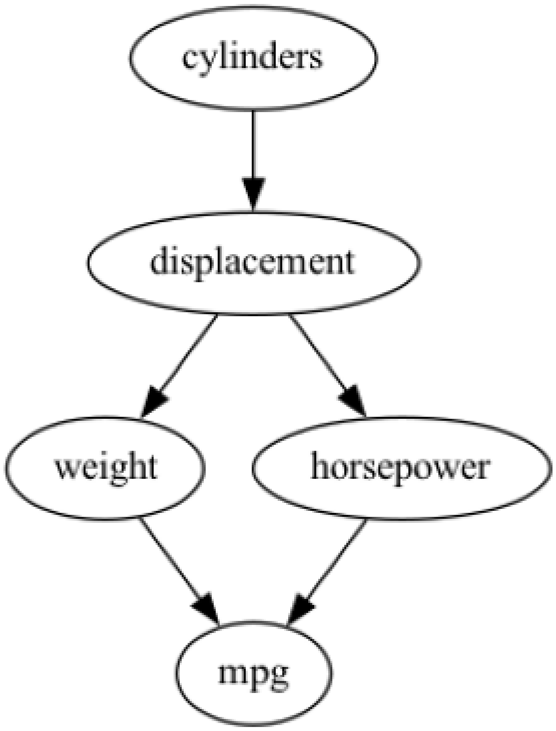
```
import pandas as pd
import networkx as nx
import numpy as np

from dowhy import gcm
from dowhy.utils.plotting import plot, bar_plot

# Load Auto MPG data: Quinlan,R.. (1993). Auto MPG. UCI Machine Learning Rep
auto_mpg_data = pd.read_csv('http://archive.ics.uci.edu/ml/machine-learning-
                            delim_whitespace=True,
                            header=None,
                            names = ['mpg',
                                     'cylinders',
                                     'displacement',
                                     'horsepower',
                                     'weight',
                                     'acceleration',
                                     'model year',
                                     'origin',
                                     'car name'])
auto_mpg_data.dropna(inplace=True)
auto_mpg_data.drop(['model year', 'origin', 'car name'], axis=1, inplace=Tru

mpg_graph = nx.DiGraph([('cylinders', 'displacement'),
                        ('cylinders', 'displacement'),
                        ('displacement', 'weight'),
                        ('displacement', 'horsepower'),
                        ('weight', 'mpg'),
                        ('horsepower', 'mpg')])

plot(mpg_graph)
```



Seeing this graph, we can expect some strong confounders between the nodes, but nevertheless, we will see that the ICC method still provides non-trivial insights.

Let's define the corresponding structural causal model and fit it to the data:

In [2]:

```
scm_mpg = gcm.StructuralCausalModel(mpg_graph)
gcm.auto.assign_causal_mechanisms(scm_mpg, auto_mpg_data)
gcm.fit(scm_mpg, auto_mpg_data)
```

```
Fitting causal mechanism of node mpg: 100%|█████| 5/5 [00:00<00:00, 12.86it/
s]
```

Optionally, we can get some insights into the performance of the causal mechanisms by using the evaluation method:

In [3]:

```
print(gcm.evaluate_causal_model(scm_mpg, auto_mpg_data, evaluate_invertibili
```

```
Evaluating causal mechanisms...: 100%|████████| 5/5 [00:00<00:00, 4132.32it/
s]
```

```
Evaluated the performance of the causal mechanisms and the overall average K
L divergence between generated and observed distribution. The results are as
follows:

==== Evaluation of Causal Mechanisms ====
Root nodes are evaluated based on the KL divergence between the generated an
d the observed distribution.
Non-root nodes are evaluated based on the (normalized) Continuous Ranked Pro
bability Score (CRPS), which is a generalizes the Mean Absolute Percentage E
rror to probabilistic predictions. Since the causal mechanisms produce condi
tional distributions, this should give some insights into their performance
and calibration. However, note that many algorithms are still relatively rob
ust against poor model performances.

--- Node cylinders: The KL divergence between generated and observed distrib
ution is 0.023328176742595987.
The estimated KL divergence indicates an overall very good representation of
the data distribution.

--- Node displacement: The normalized CRPS of this node is 0.173283950390501
98.
The estimated CRPS indicates a very good model performance.

--- Node weight: The normalized CRPS of this node is 0.17808725484841442.
The estimated CRPS indicates a very good model performance.

--- Node horsepower: The normalized CRPS of this node is 0.2073388436392234.
The estimated CRPS indicates a good model performance.

--- Node mpg: The normalized CRPS of this node is 0.2795755556077868.
The estimated CRPS indicates a good model performance.

==== Evaluation of Generated Distribution ====
The overall average KL divergence between the generated and observed distrib
ution is 0.87783329247371
The estimated KL divergence indicates a good representation of the data dist
ribution, but might indicate some smaller mismatches between the distributio
ns.

==== NOTE ====
Always double check the made model assumptions with respect to the graph str
ucture and choice of causal mechanisms.
All these evaluations give some insight into the goodness of the causal mode
l, but should not be overinterpreted, since some causal relationships can be
intrinsically hard to model. Furthermore, many algorithms are fairly robust
against misspecifications or poor performances of causal mechanisms.
```

After defining our structural causal model, we can now apply the ICC method to obtain more insights into what factors influence fuel consumption. This could help us improve the design process. Note that by default, we attribute the variance of the target node to the upstream nodes.

```
In [4]: iccs_mpg = gcm.intrinsic_causal_influence(scm_mpg, target_node='mpg')
```

```
Evaluate set function: 32it [00:00, 672.76it/s]
```

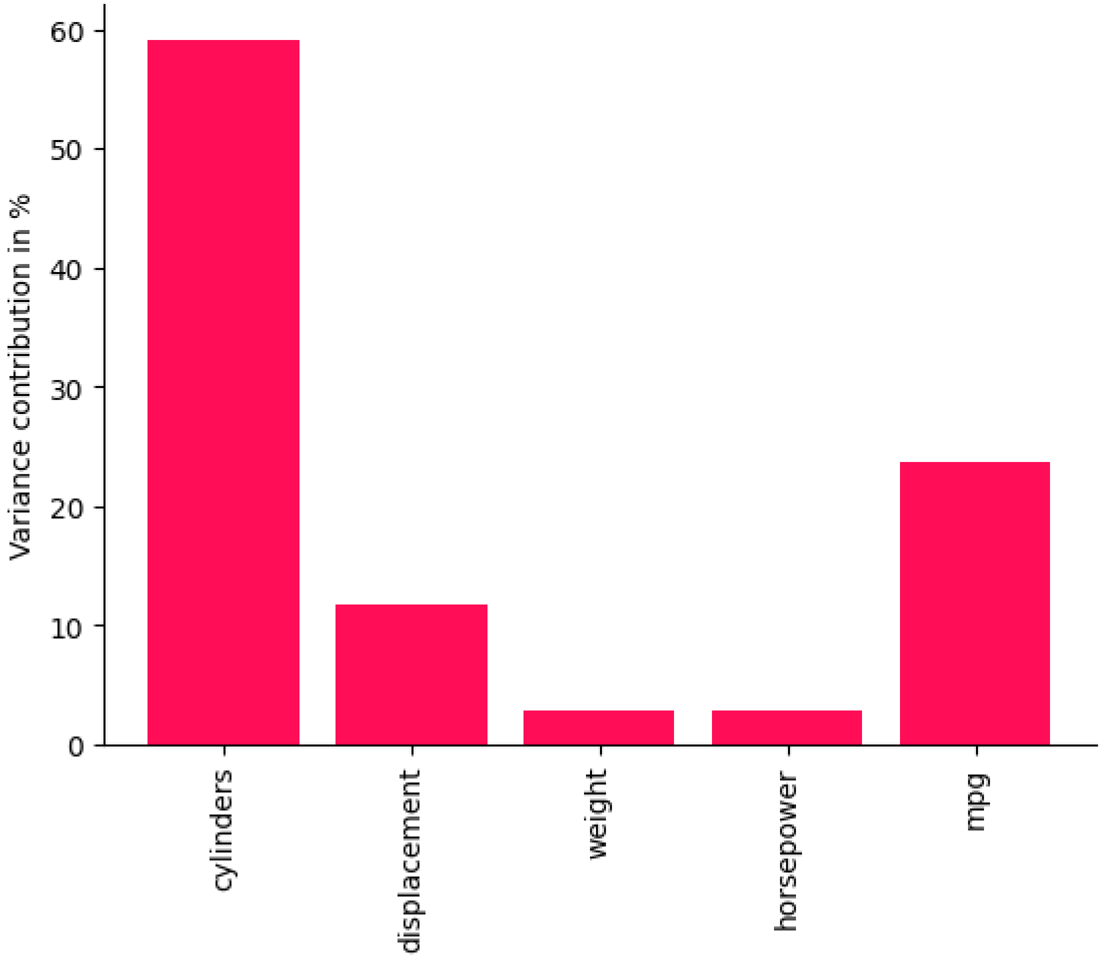
For a better interpretation of the results, we convert the variance attribution to percentages by normalizing it over the total sum.

In [5]:

```
def convert_to_percentage(value_dictionary):
    total_absolute_sum = np.sum([abs(v) for v in value_dictionary.values()])
    return {k: abs(v) / total_absolute_sum * 100 for k, v in value_dictionar
```

In [6]:

```
bar_plot(convert_to_percentage(iccs_mpg), ylabel='Variance contribution in %
```



It turns out that the number of cylinders already explains a large fraction of the fuel consumption and the intermediate nodes like displacement, horsepower, and weight mostly inherit uncertainty from their parents. This is because, although weight and horsepower are the more direct predictors of mpg, they are mostly determined by displacement and cylinders. As we also see with the contribution of mpg itself, roughly 1/4 of the variance of mpg remains unexplained by all of the above factors, which may be partially due to model inaccuracies.

While the model evaluation showed that there are some inaccuracies with respect to the KL divergence between the generated and observed distributions, we see that ICC still provides non-trivial results in the sense that the contributions differ significantly across nodes and that not everything is simply attributed to the target node itself.

> Note that estimating the contribution to the variance of the target in ICC can be seen as a nonlinear version of ANOVA that incorporates the causal structure.

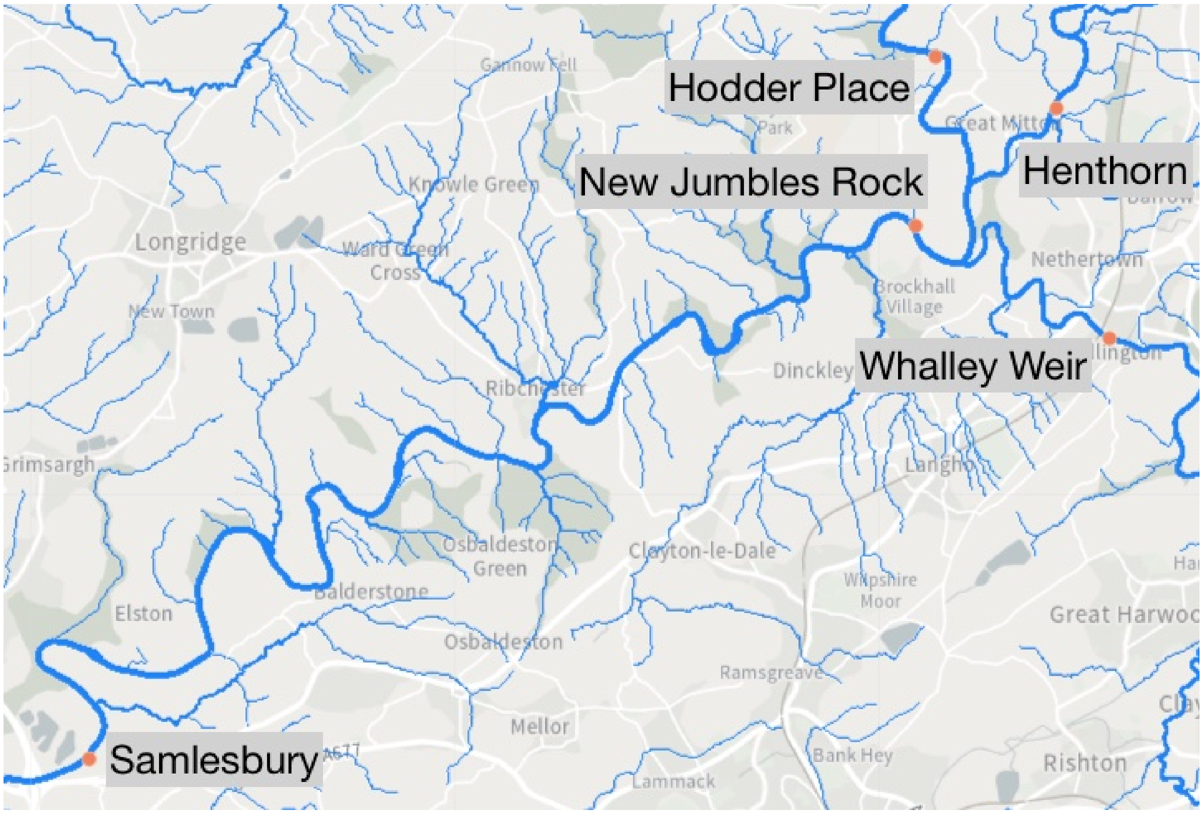
Intrinsic influence on river flow

In the next example, we look at different recordings taken of the river flows (m^3/s) at a 15 minute frequency across 5 different measuring stations in England at Henthorn, New Jumbles Rock, Hodder Place, Whalley Weir and Samlesbury. The data is taken from the UK Department for Environment Food & Rural Affairs website. Here is a map of the rivers:

In [7]:

```
from IPython.display import Image
Image('river-map.jpg')
```

Out[7]:

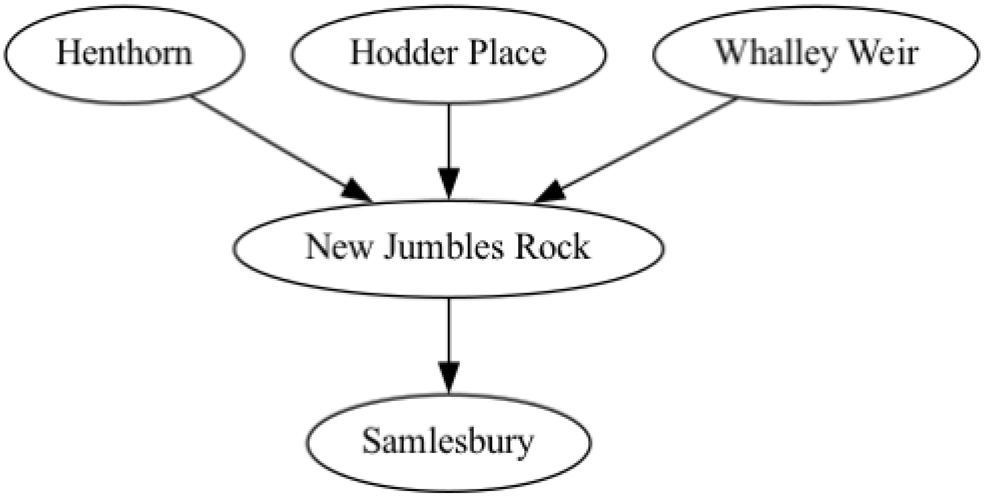


New Jumbles Rock lies at a confluence point of the 3 rivers passing Henthorn, Hodder Place, and Whalley Weir and New Jumbles Rock flows into Samlesbury. The water passing a certain measuring station is certainly a mixture of some fraction of the amount observed at the next stations further upstream plus some amount contributed by streams and little rivers entering the river in between. This defines our causal graph as:

In [8]:

```
river_graph = nx.DiGraph([('Henthorn', 'New Jumbles Rock'),
                          ('Hodder Place', 'New Jumbles Rock'),
                          ('Whalley Weir', 'New Jumbles Rock'),
                          ('New Jumbles Rock', 'Samlesbury')])
```

```
plot(river_graph)
```

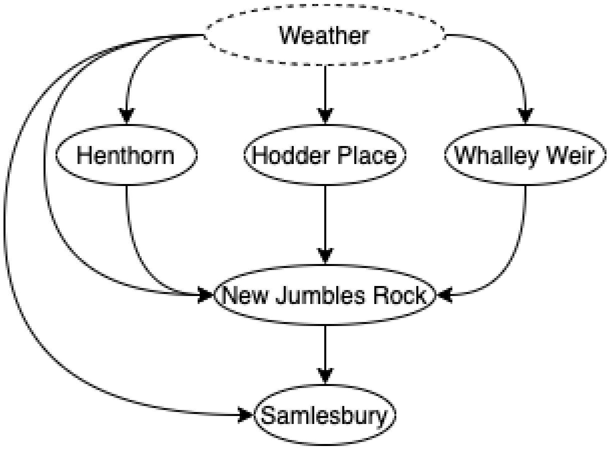


Here, we are interested in the causal influence of the upstream rivers on the Samlesbury river. For instance, to obtain a better understanding of how the river flows behave and to potentially plan mitigation steps to avoid overflows. Similar to the example before, we would expect these nodes to be heavily confounded by, e.g., the weather. That is, the true graph is more likely to be along the lines of:

In [9]:

```
Image('river-confounded.png')
```

Out[9]:



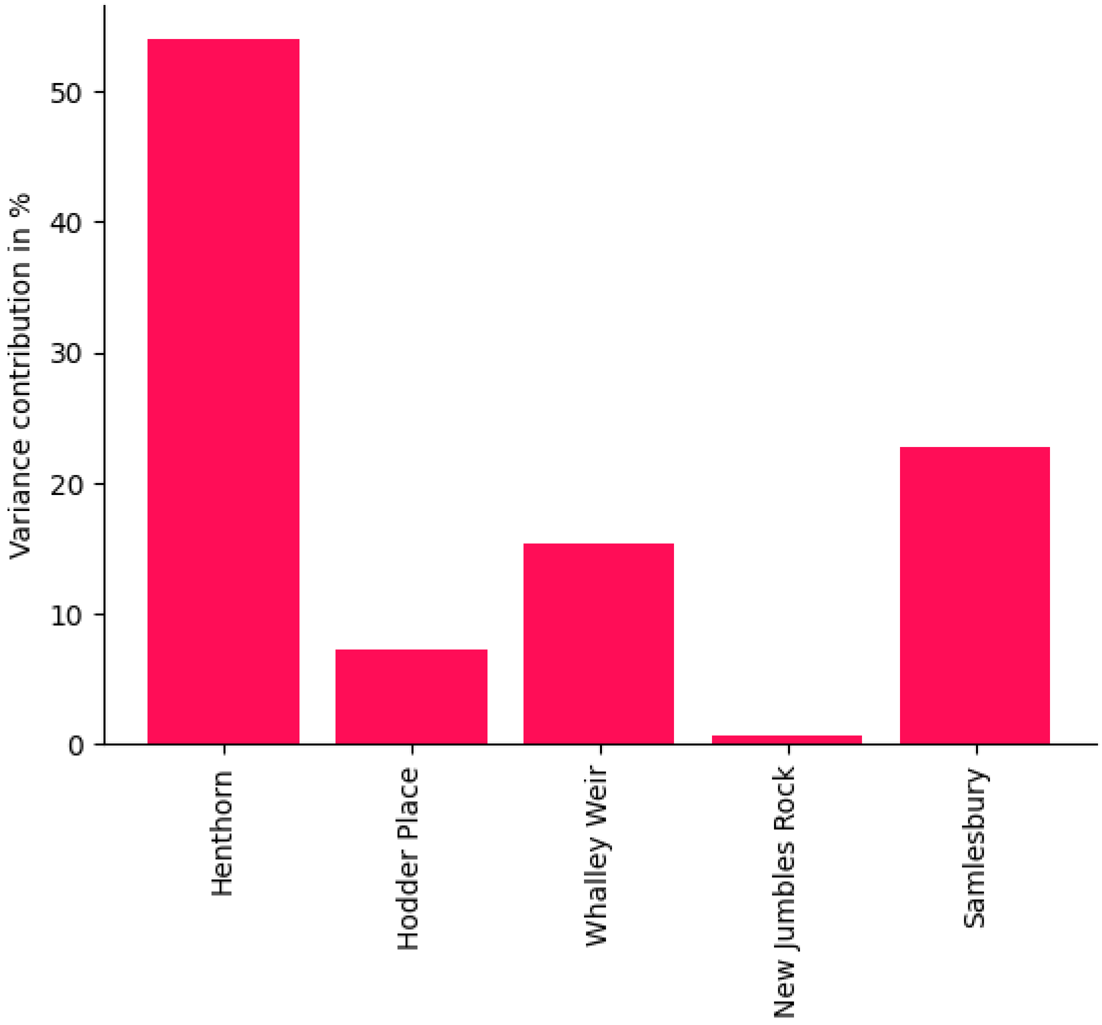
Nevertheless, we still expect the ICC algorithm to provide some insights into the contribution to the river flow of Samlesbury, even with the hidden confounder in place.

In [10]:

```
river_data = pd.read_csv("river.csv", index_col=False)

scm_river = gcm.StructuralCausalModel(river_graph)
gcm.auto.assign_causal_mechanisms(scm_river, river_data)
gcm.fit(scm_river, river_data)

iccs_river = gcm.intrinsic_causal_influence(scm_river, target_node='Samlesbu
bar_plot(convert_to_percentage(iccs_river), ylabel='Variance contribution in
Fitting causal mechanism of node Samlesbury: 100%|█| 5/5 [00:00<00:00, 158.8
5i
Evaluate set function: 32it [00:00, 1625.01it/s]
```



Interestingly, the intrinsic contribution of New Jumbles Rock on Samlesbury is small, although the interventional effect on New Jumbles Rock would certaintly have a large effect. This illustrates that ICC does not measure influence in the sense of the strength of a treatment effect and points out here that New Jumbles Rock simply passes the flow onto Samlesbury. The contribution by Samlesbury itself represents the (hidden) factors that are not captured. Even though we can expect the nodes to be heavily confounded by the weather, the analysis still provides some interesting insights which we only obtain by carefully distinguishing between influences that were just inherited from the parents and 'information' that is newly added by the node.