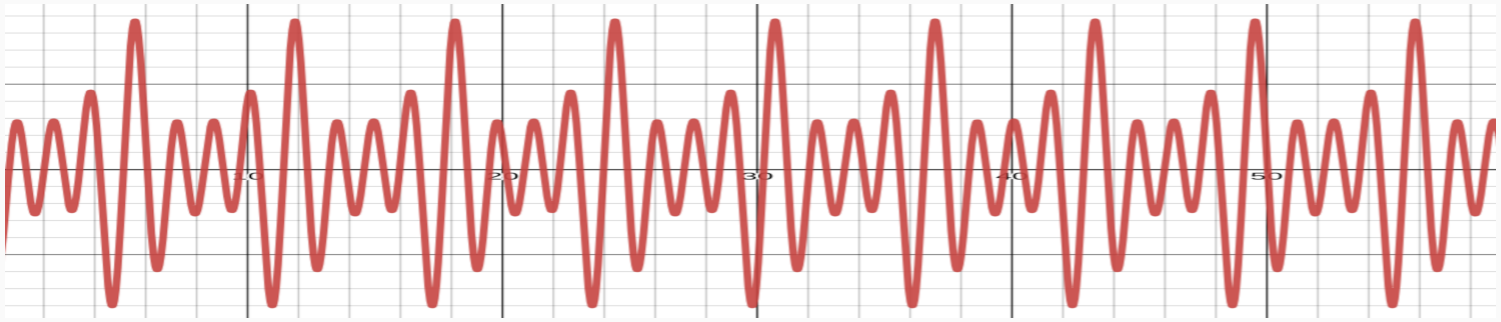
Model Validation



Prof. Fabrício Olivetti de França

Federal University of ABC

05 February, 2024

Model Validation

Model Validation

As already stressed throughout this course, there are three main approaches for nonlinear regression:

- Using an overparameterized generic model (opaque model).
- Manually crafting the nonlinear model.
- Using Symbolic Regression to find a nonlinear model with as few parameters as possible.

Model Validation

While crafting the model using first principles, you may have some properties that you want to enforce into your model, either because of some requirements or from a prior knowledge about the behavior of the system.

In this situation, the practitioner can enforce those using their own expertise.

Model Validation

For example, due to EU regulations[1], the practitioner will create a model that will allow them to debug how the output is generate in a clear manner. Also, they may want to ensure fairness in the predictions.

[1](https://www.europarl.europa.eu/news/en/press-room/20240308IPR19015/artificial-intelligence-act-meps-adopt-landmark-law)[https://www.europarl.europa.eu/news/en/press-room/20240308IPR19015/artificial-intelligence-act-meps-adopt-landmark-law]

Model Validation

This is usually a problem for opaque models that are often hard to debug and not flexible enough to enforce some propeties of interest.

In the current literature, there are some techniques that can extract information from opaque models to have a better understanding. But this may not be enough in practice.

Model Validation

With the *vanilla* symbolic regression, you have the possibility of finding a model that attends to all your requirements. To increase the probability of finding the correct model, you need at least one of these:

- Noiseless data.
- Representative data.
- Luck ☘
- A well calibrated SR algorithm.

Model Validation

With the *vanilla* symbolic regression, you have the possibility of finding a model that attends to all your requirements. To increase the probability of finding the correct model, you need at least one of these:

- Noiseless data.
- Representative data.
- Luck ☘
- A well calibrated SR algorithm.

We can only afford the last one!

Model Validation

Another important motivation for model validation is that, depending on the hyper-parameters, the SR algorithm can favor large and overparameterized models that will have a high goodness-of-fit without the remaining desider-ata.

Example of desiderata

Some example of objectives beyond the goodness-of-fit[2] are:

- The ability to understand and explain model behavior
- Scientific plausibility of the model
- Whether the model is generalizable and capable of extrapolation
- Boundedness and safe operation under all circumstances
- Efficiency of calculating predictions or computational effort required for training the model

[2] Gabriel Kronberger, Bogdan Burlacu, Michael Kommenda, Stephan M. Winkler, and Michael Affenzeller. Symbolic Regression. tbr.

Example of desiderata

Besides those, we may also want a model that:

- Ensures a fair inference to different classes of the sample.
- Behaves according to pre-established norms.

Ability to understand and explain model behavior

In the beggining of the course, it was clear that a linear model is easy to understand:

- With every unitary change in x we observe a change proportional to β in the outcome.
- Even if we have a linear model with non-linear features, they can have physical meaning. E.g., $v = s/t$, the inverse interaction of displacement and time gives us the average velocity.

Ability to understand and explain model behavior

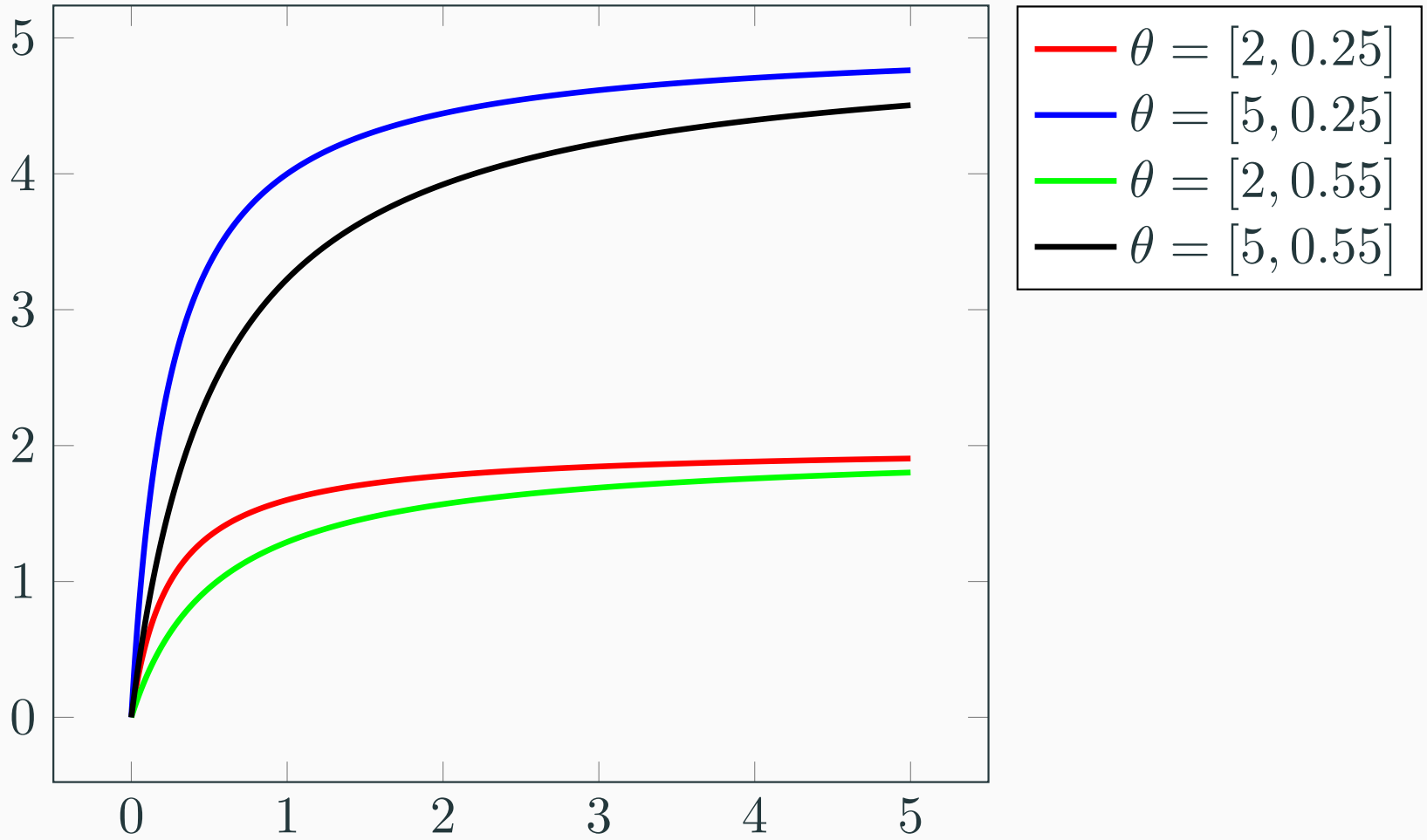
When we have a nonlinear regression model, these interepretations are not as straightforward:

$$f(x;\theta) = \frac{\theta_1 x}{\theta_2 + x},$$

The association between the input variable and the outcome is not easily understood.

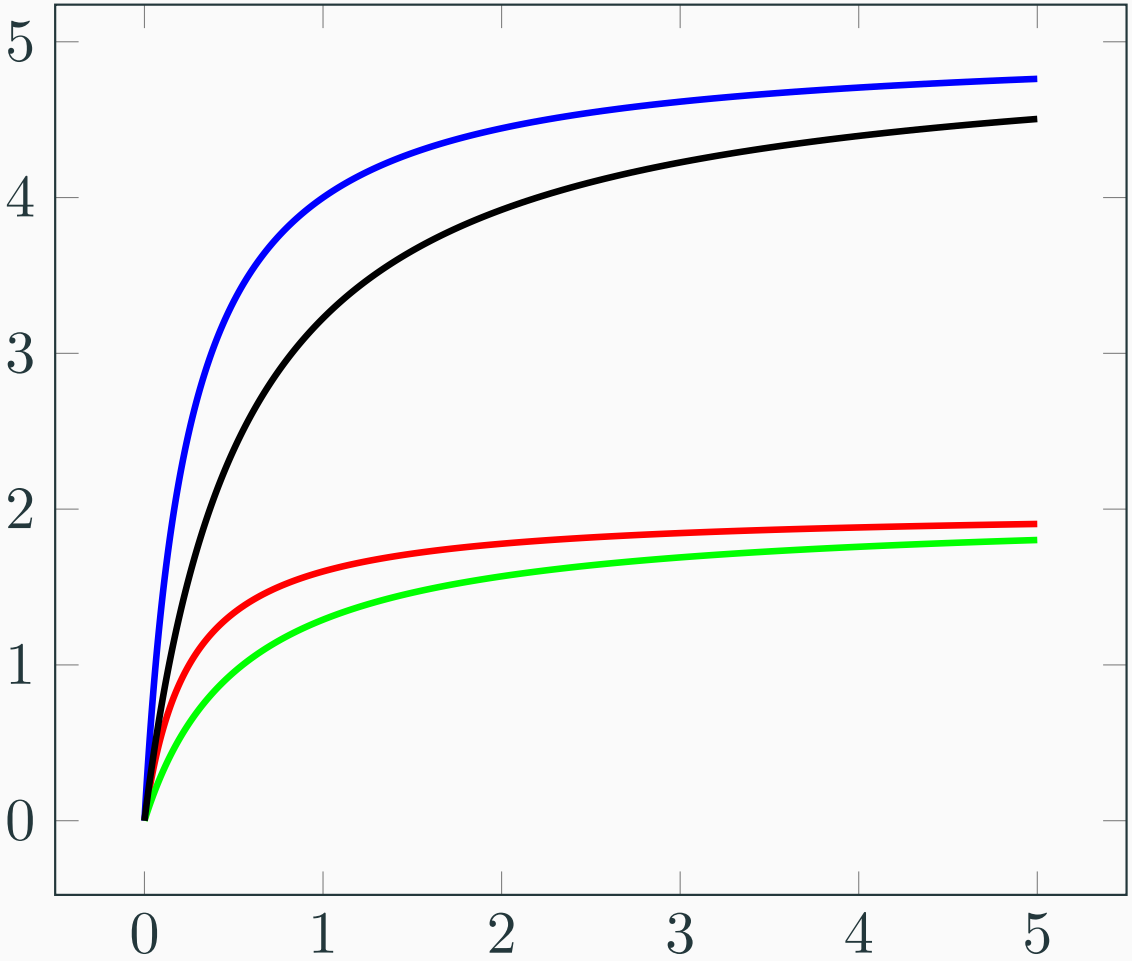
Ability to understand and explain model behavior

We can try to understand the behavior with a plot for different values of θ:



Ability to understand and explain model behavior

::::columns :::column



::: :::column - This model has a saturation value close to θ_1 - The higher the value of θ_0, the slower the speed to reach the saturation. When $x = \theta_0$

Ability to understand and explain model behavior

Having the context of the model can help gain additional insights. This particular model can represent the **Michaelis–Menten kinetics** that describes the reaction rate ($f(x;\theta)$) to the concentration of a substrate (x).

Knowing the phyisical meanings of θ will give us insight when fitting this model for different enzymes.

We can see that, once we contextualize the model and add expert knowledge, we can gain insights from nonlinear models as well, as long as their parameters are meaningful in our context (thus, minimize the number of parameters is desired).

Ability to understand and explain model behavior

In short, inspecting the model for the ability of understanding and explaining can be done by:

- Contextualizing the model
- Applying expert knowledge
- Plotting the behavior of the function with different parameter values

Additional tools will be given in later lectures when we talk about explainability.

Scientific plausibility of the model

Related to the previous desiderata, scientific plausibility refers to whether the model:

- Behaves similarly to the observed phenomena.
- Is correct w.r.t. a dimensional analysis (or whether all meta-features are dimensionless)
- Possesses a phyisical meaning
- Does not misbehave

This can be inspect through visual plots and expert knowledge.

Whether the model is generalizable and capable of extrapolation

The SR model is fitted on a limited data set that not necesseraly captures a

Boundedness and safe operation under all circumstances

Efficiency of calculating predictions or computational effort required for training the model

Ensures a fair inference to different classes of the sample.

Behaves according to pre-established norms.

Visual tools

The plots regarding the predictions and residuals can be insightful and provide a tool for model inspection. From these plots we can understand whether the model meets our expectations and whether there is any unexpected behavior.

Noise variance plot

One example of a plot is the predicted values against the dependent variable as observed in the data. To illustrate this and the next plots, we will fit our simulated grade dataset with PyOperon, PySR and TIR:

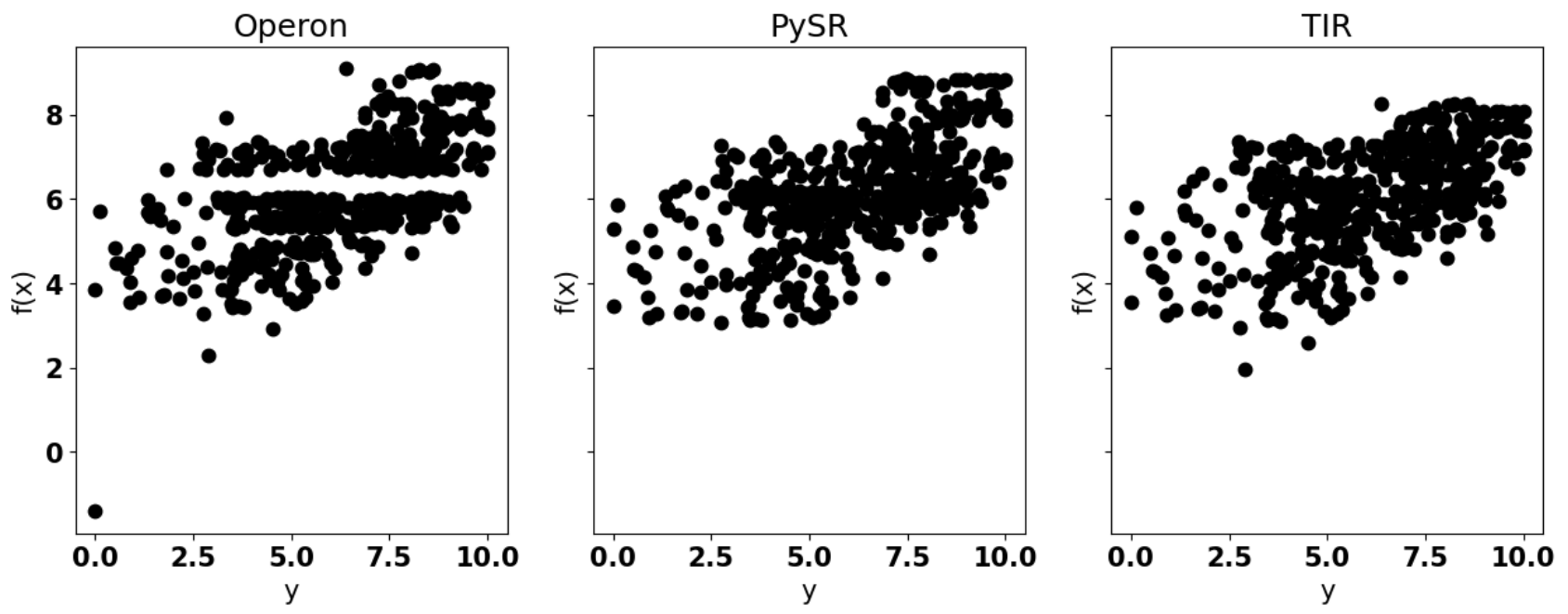
```
regs = [SymbolicRegressor(),
        TIRRegressor(100, 100, 0.3, 0.7, (-3, 3), transfunctions='Id',
        PySRRegressor(binary_operators=["+", "*"], unary_operators=[])
       ]
for i in range(3):
  regs[i].fit(x.reshape(-1,1),y)
```

Noise variance plot

We can check the noise variance with:

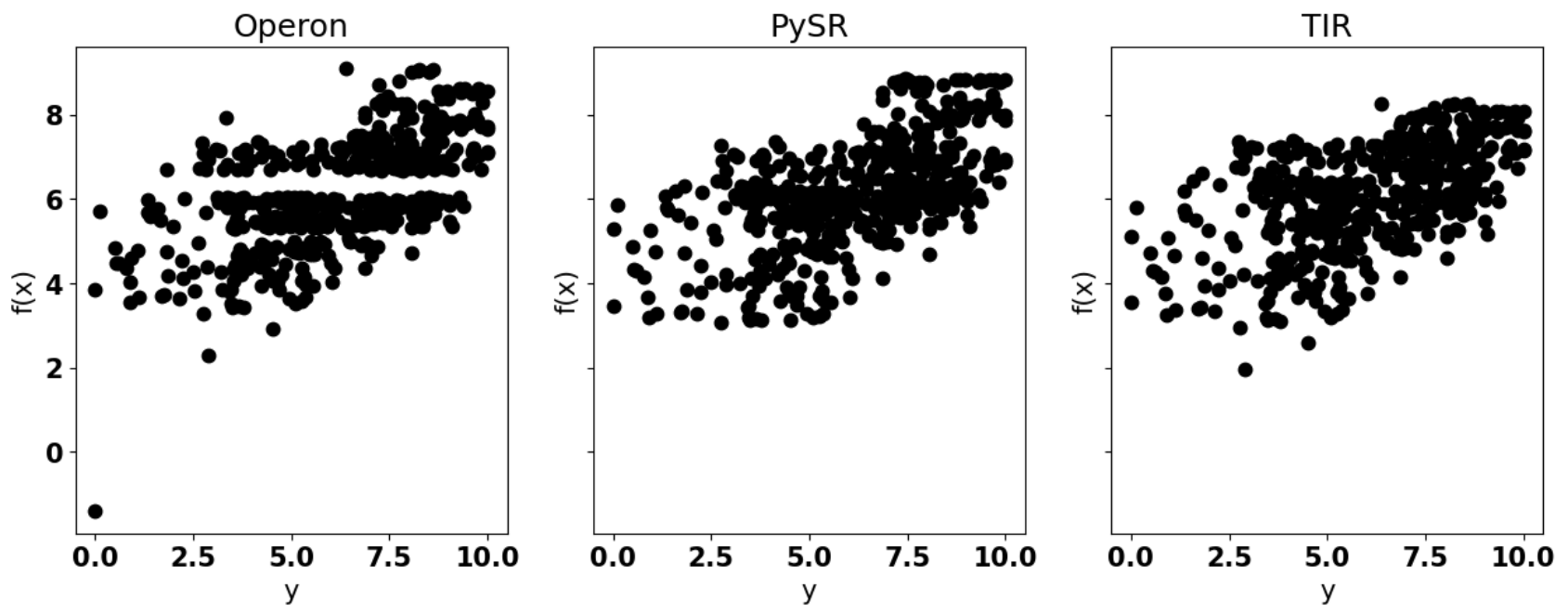
```
_,axs = plt.subplots(1,3, figsize=(15,5), sharey=True)
name = ['Operon', 'PySR', 'TIR']
for i in range(3):
  axs[i].plot(y, regs[i].predict(x.reshape(-1,1)), '.', color='black',
  axs[i].set_xlabel('y')
  axs[i].set_ylabel('f(x)')
  axs[i].set_title(name[i])
```

Noise variance plot



A perfect model would have all the points in the 45 degrees diagonal.

Noise variance plot



We can see from these plots that none of the models returns a satisfactory result. Also, we can see that all of them have a bias in mispredicting grades below 5 (usually for a higher grade).

Q-Q plot

Another important plot is the quantile-quantile plot (Q-Q plot) that plots the assumed error distribution of the data matches the distribution of the residuals of the model.

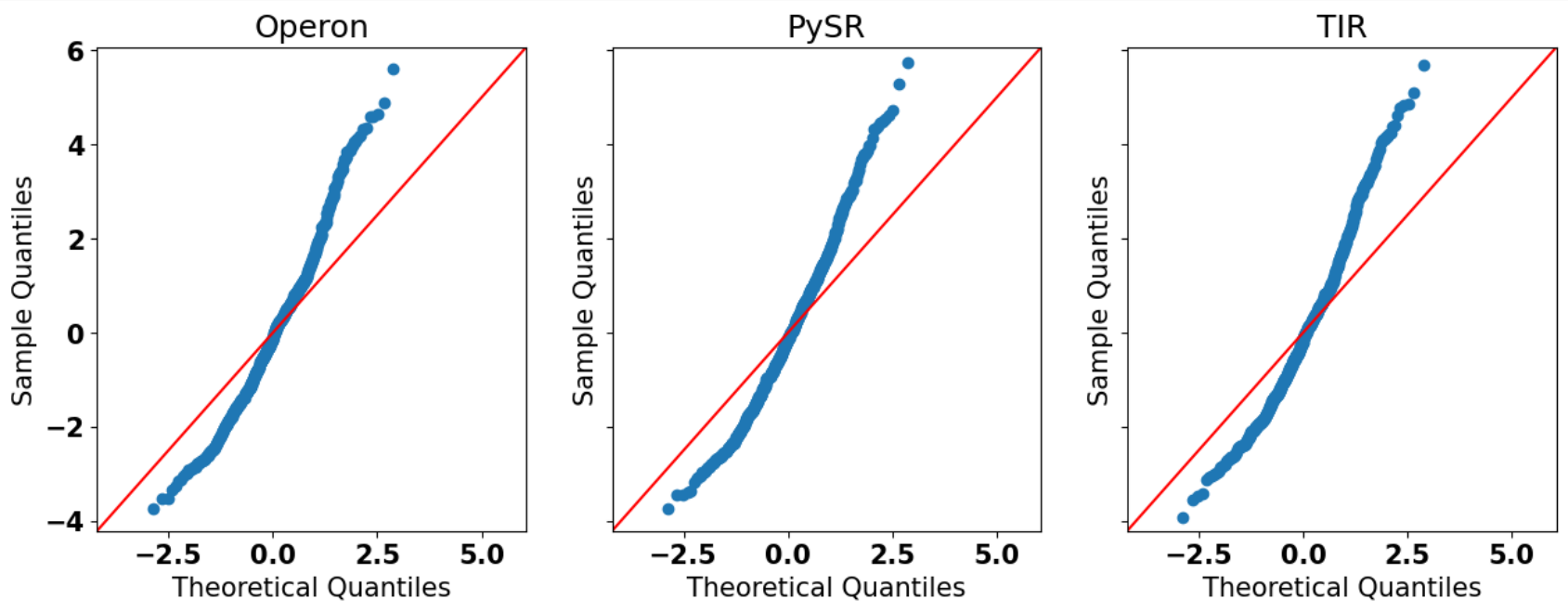
To make the Q-Q plot, we calculate the residuals of our model, sort them in increasing order, and plot each point against the inverse of the cumulative density function of the assumed distribution.

Q-Q plot

(qqplot assumes normal distribution as the default)

```
1  import statsmodels.api as sm
2
3  _,axs = plt.subplots(1,3, figsize=(15,5), sharey=True)
4
5  for i in range(3):
6    sm.qqplot(regs[i].predict(x.reshape(-1,1))[:,0]-y, line ='45', ax=axs
7    axs[i].set_title(name[i])
```

Q-Q plot



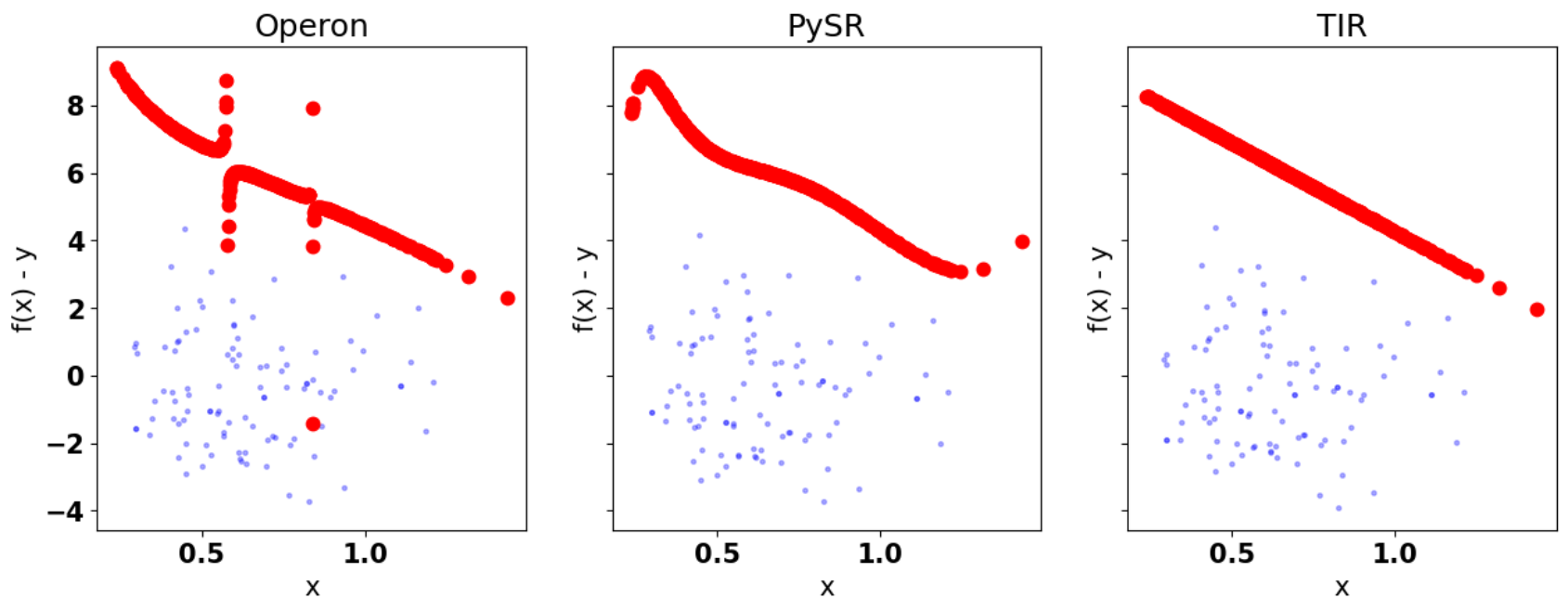
We can see from these plots that none of the models matches the expected distribution for the residuals.

Residuals plot

Another interesting plot is the residuals plots in which we plot a choice of x_i against $f(x)$ and the residuals:

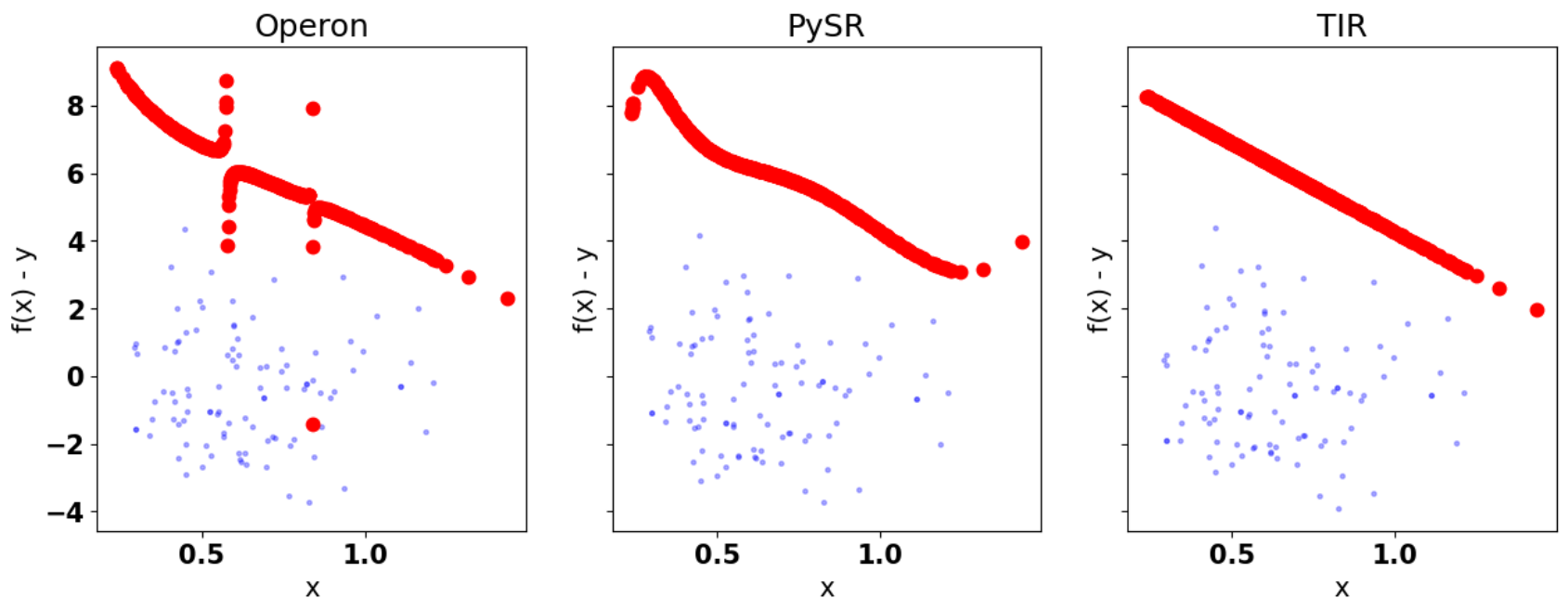
```
import statsmodels.api as sm

_,axs = plt.subplots(1,3, figsize=(15,5), sharey=True)

for i in range(3):
  axs[i].plot(x, regs[i].predict(x.reshape(-1,1))[:,0] - y, '.', color=
  axs[i].plot(x, regs[i].predict(x.reshape(-1,1)), '.', color='red', ma
  axs[i].set_xlabel('x')
  axs[i].set_ylabel('f(x) - y')
  axs[i].set_title(name[i])
```

Residuals plot



These plots show that all of these models have an error ranging from -4 to 4 but mostly concentrated on neagtive residuals. This means it tends to understimate the true value.

Residuals plot



Also, we can see that Operon created a model with some discontinuities (possibily because of division) and TIR chose a linear model.

Next lecture

- Model Selection

Acknowledgments