ENABLING EQUATION LEARNING WITH BAYESIAN MODEL EVIDENCE VIA SYSTEMATIC R^2-ELIMINATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep learning is a powerful method for tasks like predictions and classification but lacks interpretability and analytic access. Instead of fitting up to millions of parameters, an intriguing alternative for a wide range of problems would be to learn the governing equations from data. The resulting models would be concise, parameters could be interpreted, the model could adjust to shifts in data, and analytic analysis would allow for extra insights. Common challenges are model complexity identification, stable feature selection, expressivity, computational feasibility, and scarce data. In our work, the mentioned challenges are addressed by combining existing methods in a novel way. We choose multiple regression as a framework and argue that a large space of model equations is captured. For feature selection, we exploit the computationally cheap coefficient of determination (R^2) for a model elimination process in a semi-comprehensive search. Final model selection is achieved by exact values of the Bayesian model evidence with empirical priors, which is known to identify suitable model complexity without relying on mass data. Random polynomials, an epidemiological model, and the Lorenz system are used as examples. For the Lorenz system, which is particularly challenging due to its chaotic nature, we demonstrate the favourable performance of our approach to existing state-of-the-art like SINDy.

1 INTRODUCTION AND RELATED WORK

The arguably best way to learn about a system is to uncover its underlying laws. This principle is the essence of the natural sciences. In the mathematical sciences, these laws are typically represented in form of governing equations. With these equations at hand, predictions can be made, and new insights can be distilled analytically.

A hall mark of such a mathematical description are concise equations that enable analytic work. Naturally, using parsimonious models, often only the main tendency of made observations can be captured. But it is these main tendencies that enable us to learn about a system.

The above principle has motivated scientists to infer the governing equations of a system directly from the observations made on the system. Such endeavours go back as far as to the year 1609 when Johannes Kepler published his first laws of planetary motion he deduced from observations Donahue & Gingerich (1992), up until today where with modern machine learning significant progress has been made in automated inference of equations from data.

This discipline of machine learning is often referred to as *equation learning* or *symbolic regression*. Compared to deep learning, the focus shifts from predictions to learning a model that maximizes expressivity while parsimoniously minimizing model size enabling interpretability. This delicate balance between interpretability and expressivity is the central challenge of equation learning. Other challenges are stable feature selection, computational feasibility, and dealing with scarce data.

Previous works addressing these challenges roughly split into three approaches: evolutionary algorithms, trees and neural network representations, and regularized regression.

A popular example of evolutionary algorithms is EUREQA, a commercial software for symbolic regression Dubčáková (2011); Stoutemyer (2013). Tree representations build equations by combining basic operations, which is then used to minimize regularized objective functions Vaddireddy et al.

(2020), to define a posterior sampler with sparsity promoting priors Jin et al. (2019), or minimized using mixed-integer linear programming Neumann et al. (2020). Similar to expression trees, neural network architectures have been used to represent equations, where the nodes of the network are replaced by building blocks of expressions Martius & Lampert (2016); Sahoo et al. (2018); Werner et al. (2021). These neural networks are then trained with regularized minimizers applied to objective functions Rackauckas et al. (2020). In a similar direction, a more flexible approach allowing to take physics-informed properties like symmetries into account has been developed Udrescu & Tegmark (2020).

Applications of these methods can be found in complementing deep neural networks to improve generalisation Arabshahi et al. (2018) or for reducing the amount of data required for training Yang et al. (2021). Other applications can be found for spatio-temporal biological data Nardini et al. (2020) or uncovering complex ecosystem dynamics Chen et al. (2019).

These approaches are highly non-linear and require advanced optimisation algorithms. A typical, simpler solution are linear regression models, where the features are replaced by basis functions on observed data. Regularized regression then ensures sparse estimates for the weights on the basis functions Hastie et al. (2009), which translate into concise mathematical expressions. A prominent and widely used method for sparse regression is the LASSO using ℓ_1-regularization Tibshirani (1996). The benefit of ℓ_1-regularization is that it leads to convex objective functions, for which efficient optimisers exist. Recent advancements of the LASSO are Zheng et al. (2019); Tibshirani & Friedman (2020). However, as we will also see later, while sparse regression works well with independent features for reconstruction tasks, the correlations introduced by basis function expansion can lead to detrimental instability for equation learning.

More successful for equation learning are therefore ℓ_0-regularized objective functions, which, however, require specialised optimisers. One of the leading approaches utilising ℓ_0-regularization is SINDy (Sparse Identification of Nonlinear Dynamics) Brunton et al. (2016); de Silva et al. (2020). Many extensions to SINDy have been made, e.g. to partial differential equations Rudy et al. (2017), improved noise robustness through automated differentiation Kaheman et al. (2020), supplemented with a-posteriori (MAP) estimates Niven et al. (2020), re-weighted ℓ_1-regularization Cortiella et al. (2021) and more flexible regularization Champion et al. (2020). In Bayesian linear regression, sparsity promoting priors replacing the ℓ_0-regularization have been used for equation learning Nayek et al. (2021), as well as threshold sparse regression Zhang & Lin (2018; 2021). Restricting basis functions to quadratic order, the linear structure of the models allow for deterministic results even in case of the more difficult ℓ_0-regularization Schaeffer et al. (2018).

In our work, we address the mentioned challenges of equation learning by fully exploiting the linearity of the regression models without introducing any regularization. Our strategy is fundamentally different to all mentioned approaches mentioned above, and, to our knowledge, has not yet been explored in the literature. To ensure the sparsity required for concise model equations, we use the Bayesian model evidence, which is proportional to the probability of the model being the true model for the data Murphy (2012). Maximising the evidence in model space hence dismisses all models with more detail than there is evidence for in the data. These properties make the evidence a perfect criterion to penalise overfitting, and thus targeting the challenge of finding the best balance between interpretability and expressivity Höge et al. (2018). Choosing specially tuned conjugate priors, we can use analytic expressions for the evidence.

Despite the analytically known model evidence, it is still impossible to loop through candidate models that can be built from the basis function expansion. We therefore first apply an elimination process that only requires the coefficient of determination, R^2. This R^2-elimination process takes place in a semi-comprehensive search (SCS), where all models that can be built from the basis functions are tested or excluded by greedy steps. Using R^2 in this way to enable Bayesian model selection for equation learning, it turns out that our SCS strategy allows to identify the correct model size just from using R^2, despite its non-existing overfitting penalty.

To test our approach, we use artificially generated data from known models.

2 LINEAR EQUATION LEARNING

In this section, we show how linear regression can be applied to equation learning, which will set the basis of this work. Details to regression can be found in Montgomery et al. (2012).

Starting point are N observations (y_i, x_{ij}), where the index i denotes data points, and the index j denotes features. We assume that the response (dependent) variable y is given as a function of explanatory (independent) variable $\boldsymbol{x}$,

$$y = f(\boldsymbol{x}) + \sigma z, \tag{1}$$

where $f(\boldsymbol{x})$ defines the model, z is a standard normal random variable, and σ^2 is the variance of the noise term. The explanatory observations are often organized in terms of a design matrix $\boldsymbol{X}$, with features in columns and datapoints in rows, $X_{ij} = x_{ij}$.

We assume that $f(\boldsymbol{x})$ can be given in terms of p basis functions $k_n(\boldsymbol{x})$,

$$f(\boldsymbol{x}) = \sum_{n=1}^{p} w_n k_n(\boldsymbol{x}), \tag{2}$$

with weights w_n, known as basis function expansion. Similar to the design matrix, we can define a basis function design matrix $\boldsymbol{K}$ with elements $\boldsymbol{K}_{in} = k_n(\boldsymbol{x}_i)$.

The ordinary least squares (OLS) estimates for w_n are known to be

$$\hat{\boldsymbol{w}} = (\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K})^{-1}\boldsymbol{K}^{\mathrm{T}}\boldsymbol{y} \tag{3}$$

Predictions based on the OLS estimates are then given by $\hat{\boldsymbol{y}} = \boldsymbol{K}\hat{\boldsymbol{w}}$.

From the normality of linear regression, it is known that the estimates $\hat{\boldsymbol{w}}$ follow a normal distribution with the mean given by the true values for $\boldsymbol{w}$ and the variance given by $\hat{\sigma}^2 = \frac{(\boldsymbol{y}-\hat{\boldsymbol{y}})^{\mathrm{T}}(\boldsymbol{y}-\hat{\boldsymbol{y}})}{N-p}$. We will use these properties later to empirically define the prior in the Bayesian description.

In view of (3), once a choice of $k_n(\boldsymbol{x})$ is made, the actual learning of the model is straight forward. The difficult part is the choice of $k_n(\boldsymbol{x})$: on the one hand we require sufficient expressivity of the model to minimize bias, on the other hand we want to avoid overfitting to minimize variance of predictions. This bias-variance tradeoff essentially dictates the number of $k_n(\boldsymbol{x})$, i.e. the effective dimension of feature space or complexity. Apart from the appropriate model size, we also seek the "correct" $k_n(\boldsymbol{x})$, in the sense that the true $f(\boldsymbol{x})$ is recovered from data generated by (1).

A common approach to avoid overfitting is regularization,

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}\in\mathbb{R}^M}{\arg\min}\left[\left\|\boldsymbol{K}\boldsymbol{w}-\boldsymbol{y}\right\|_2^2 + \lambda\|\boldsymbol{w}\|_q\right], \tag{4}$$

where $\|\boldsymbol{w}\|_q = [\sum_n |w_n|^q]^{1/q}$, and λ is the Lagrange parameter that sets the strength of the ℓ_q penalty. The standard, sparsity promoting choice is $q = 1$ which is known as the LASSO.

2.1 COEFFICIENT OF DETERMINATION

A standard measure for goodness of fit is the coefficient of determination, R^2, which relates the variance explained by the prediction $\hat{\boldsymbol{y}}$ to the variance of the response variable y. Assuming standardised y, R^2 can be written as

$$R^2 = \frac{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}\left(\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K}\right)^{-1}\boldsymbol{K}^{\mathrm{T}}\boldsymbol{y}}{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}}. \tag{5}$$

For sparse weight estimates $\hat{\boldsymbol{w}}$, R^2 is extremely efficient to compute. A value of R^2 close to 1 signifies good predictions $\hat{\boldsymbol{y}}$.

2.2 BAYESIAN MODEL EVIDENCE

For the purpose of adequate model selection, it is helpful to formulate regression in a Bayesian setting. The main step to this end is defining the likelihood distribution for y. In the simple case of (1), y is normally distributed,

$$p_{\mathrm{li}}(y|\boldsymbol{w},\sigma,\mathcal{M}) = \mathcal{N}(\boldsymbol{\mu},\sigma), \tag{6}$$

where the mean vector is given by $\mu_i = f(\boldsymbol{x}_i)$, and we include the dependence on the choice of model $\mathcal{M}$ given by the representation (2) of $f(\boldsymbol{x}_i)$ in terms of $\boldsymbol{K}$.

Marginalising over $\boldsymbol{w}$ and σ,

$$p(y|\mathcal{M}) = \int \mathrm{d}\sigma \int \mathrm{d}\boldsymbol{w}\; p_{\mathrm{li}}(y|\boldsymbol{w}, \sigma, \mathcal{M})\, p_{\mathrm{pr}}(\boldsymbol{w}, \sigma), \tag{7}$$

we obtain the Bayesian model evidence (or marginal likelihood), which is proportional to the probability of the model $\mathcal{M}$ being the true model for the data $(\boldsymbol{y}, \boldsymbol{X})$ Murphy (2012).

The integral in (7) is typically not solvable analytically, and also poses a particularly tough numerical challenge Von Der Linden et al. (1999); Knuth et al. (2015). Fortunately, by choosing $p_{\mathrm{pr}}(\boldsymbol{w}, \sigma)$ conjugate to $p_{\mathrm{li}}(y|\boldsymbol{w}, \sigma, \mathcal{M})$, the integral becomes solvable analytically. The conjugate prior, however, is not necessarily the sensible choice from the inference point of view. In fact, making a good choice for the prior is a much debated problem Fortuin (2021). Here, we demonstrate that for the purpose of linear equation learning, the conjugate prior is a suitable choice, if hyper-parameters are distilled from data. The question whether other choices for the prior would perform significantly better is left for future research.

The conjugate prior for the likelihood (6) is the gamma-normal distribution O'Hagan & Kendall (1994)

$$p(\boldsymbol{w}, \tau|\boldsymbol{\mu}, \boldsymbol{M}, k, \vartheta) = \frac{\sqrt{\det \boldsymbol{M}}}{(2\pi)^{p/2}\Gamma(k)\vartheta^k}\; \tau^{p/2+k-1}\; e^{-\frac{\tau}{2}\,(\boldsymbol{w}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{M}(\boldsymbol{w}-\boldsymbol{\mu})-\tau/\vartheta} \tag{8}$$

with mean vector $\boldsymbol{\mu}$ and precision matrix $\boldsymbol{M}$ for the weights $\boldsymbol{w}$, and shape k and scale ϑ for the precision $\tau = 1/\sigma^2$. Plugging (8) and (6) into (7) and performing the integration, we obtain for the log-evidence per data-point the closed expression

$$\begin{aligned}\frac{1}{N}\ln p(y|\mathcal{M}) = \frac{1}{2N}\ln\frac{\det \boldsymbol{M}}{\det \boldsymbol{A}} - \frac{1}{2}\ln 2\pi - \Big(\frac{1}{2}+\frac{k}{N}\Big)\ln\Big(\frac{\xi}{2}+\frac{1}{\vartheta}\Big) \\ - \frac{k}{N}\ln\vartheta + \frac{1}{N}\ln\Gamma\Big(\frac{N}{2}+k\Big) - \frac{1}{N}\ln\Gamma(k)\end{aligned} \tag{9}$$

with $\boldsymbol{A} = \boldsymbol{K}^{\mathrm{T}}\boldsymbol{K} + \boldsymbol{M}$, $\boldsymbol{b} = \boldsymbol{K}^{\mathrm{T}}\boldsymbol{y} + \boldsymbol{M}\boldsymbol{\mu}$, and $\xi = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{y} + \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{\mu} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{b}$. This is a known result in Bayesian linear regression O'Hagan & Kendall (1994), but, to our knowledge, has not yet been utilised for equation learning. For the benefit of the reader, we detail the calculations in appendix B.

Owing to the normality of linear regression, and from standardising the data, it is reasonable to assume the following parameters for the prior: For the mean vector, we choose $\boldsymbol{\mu} = \hat{\boldsymbol{w}}$, and the precision matrix is $\boldsymbol{M}$ is taken to be diagonal with elements $\mathrm{diag}(\boldsymbol{M}) = \frac{1-p/N}{\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}-\hat{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{K}^{\mathrm{T}}\boldsymbol{y}}$, resulting in normal distributions broadened by a factor N to make the prior more uninformative. The gamma distribution entering (8) has the mode $(k-1)\vartheta$, which we set to 1 due to standardised y. The scale is set to $\vartheta = 1/2$ which appears to be broad enough for an uninformative prior.

2.3 Other selection criteria

For completeness, we mention a few more selection criteria used for comparison in this work. The adjusted R^2 Montgomery et al. (2012)

$$R^2_{\mathrm{adj}} = 1 - \frac{N-1}{N-p-1}\,(1-R^2) \tag{10}$$

equips the usual R^2 with an overfitting penalty. The Akaike information criterion (AIC) measures the loss of information by using the inferred model instead of the (unknown) true model Murphy (2012), and similarly but derived from the model evidence (7) in the big data limit, follows the Bayesian (Schwarz) information criterion,

$$\mathrm{AIC} = 2p_{\mathrm{li}}(y|\hat{\boldsymbol{w}}, \hat{\sigma}) - 2p\;, \quad \mathrm{BIC} = p_{\mathrm{li}}(y|\hat{\boldsymbol{w}}, \hat{\sigma}) - 2p\ln N \tag{11}$$

Algorithm 1 Basis function design matrix

1: Function DESIGNMATRIX($\boldsymbol{X}$)
2: **Input:** data $\boldsymbol{X}$ with N datapoints, l features
3: **Parameters:** maximum degree M_1 for individual features, maximum degree M_2 for term
4: build all powers $x_{ij}^{m_j}$, $m_j = (1, ..., M_1)$ # pre-computed for speed, limited to power M_1
5: initialise counter $p = 0$ # counts generated features form basis function expansion
6: **for** all unique l-tuples $(m_1, ..., m_l)$ **do**
7: **if** $\sum_j m_j \leq M_2$ **then** # ensure that collective power is limited to M_2
8: $p := p + 1$
9: $\boldsymbol{K}_{:,p} := \prod_j \boldsymbol{X}_{:,j}^{m_j}$ # Design matrix, candidate models in columns
10: **end if**
11: **end for**
12: **Return:** Design matrix $\boldsymbol{K}$, shape $N \times p$

3 DYNAMICAL SYSTEMS

Apart from the equations that can directly be written in the form of (2), a prominent application of linear equation learning is the sparse identification of dynamical systems,

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t)), \tag{12}$$

where $\dot{x}(t)$ denotes the time derivative of $x(t)$. To map this problem to (2), the response variable can be computed from finite differences $y_i = \frac{x_{i+1} - x_i}{\Delta t}$ for a fixed time step Δt.

3.1 MODEL CLASS

A restriction for the regression models to stay linear in its parameters is that the parameters of basis functions only enter as weights w. Basis functions like

$$e^{ax},\ \ln(a + x),\ \cos ax,\ x^a,\ \frac{1}{(a + x)^m},\ \ldots \tag{13}$$

with internal parameter a are not suitable.

This restriction might seem quite limiting. On the other hand, the function $f(\boldsymbol{x}(t), \boldsymbol{w})$ defining a dynamical system typically is linear in its parameters $\boldsymbol{w}$. The reason for that is that functions shown in (13) often reproduce when differentiated, which can be used to eliminate these functions, retrieving the standard form shown in (2) and (1).

In general, by considering the differentiated response variable,

$$y \ \mapsto\ \frac{\mathrm{d}y}{\mathrm{d}x} \simeq \frac{y(x_{i+1}) - y(x_i)}{x_{i+1} - x_i}, \tag{14}$$

if necessary to higher order, we can learn a surprisingly broad class of equations relating y and $\boldsymbol{x}$, even relations that do not exist in closed form.

4 SEMI-COMPREHENSIVE SEARCH WITH R^2

To our knowledge, all equation learning approaches include an optimisation step in various representation spaces of equations. Here, we propose a strategy that does without any numerical optimisation algorithms, and instead considers candidate models individually. Since already small dictionaries of basis functions can lead to tremendous numbers of candidate models, a combination of cheap selection criteria and successive reduction of model space with a suitable stopping criterion is required. We demonstrate how the simple criterion R^2 can be used for such a semi-comprehensive search.

In a first step, a dictionary of basis functions is generated using Algorithm 1. These basis functions consist of all possible products of available features x_j. In these products, the factors are raised to all possible combinations of powers (line 9), where we restrict individual powers to M_1 (line 4) and the

Algorithm 2 Model ranking R^2

1: Function TOPRSQ($\boldsymbol{y}, \boldsymbol{K}$)
2: **Input:** response data $\boldsymbol{y}$, design matrix $\boldsymbol{K}$
3: **Parameters:** number r of terms and t of top models
4: initialize criterion $\boldsymbol{c}$ # flexible length, to store R^2 for candidate models
5: initialize model indices $\boldsymbol{M}$ # p rows indicating terms part of models, flexible number of columns
6: initialize model number $i = 0$
7: **for** all $\boldsymbol{n} = (n_1, ..., n_p)$ with $n_j \in \{0, 1\}$, $\sum_j n_j = r$ **do** # possible selections of terms, for fixed r
8: $i := i+1$, append $\boldsymbol{c}_i$ and $\boldsymbol{M}_{:,i} = 0$
9: reduce $\boldsymbol{K}_{\text{red}} := \boldsymbol{K}_{:,n}$ # extract design matrix for selected terms
10: determine R^2 for $\boldsymbol{K}_{\text{red}}$ and $\boldsymbol{y}$ from (5)
11: store $c_i := R^2$ and $\boldsymbol{M}_{:,i} := \boldsymbol{n}$ # each column of M indicates a model with R^2 value c_i
12: **end for**
13: sort columns of $\boldsymbol{M}$ and $\boldsymbol{c}$ according to $\boldsymbol{c}$ (descending) # first columns of M now indicate top models in terms of R^2
14: **Return:** top models $M_{:,:t}$ (shape $p \times t$), criterion $c_{:t}$ (length t) # only return the t best models

combined power of a term to M_2 (line 7). For example, for $M_1 = 3$ and $M_2 = 5$, the term $x_1^4 x_3$ would not be allowed since $4 > M_1$, and the term $x_1^2 x_2 x_3^3$ would not be allowed because $2+1+3 > M_2$.

The SCS strategy we propose is based on this basis function expansion and described by Algorithm 3. It begins by considering all regression models with r non-zero weights w_j (line 9), c.f. equation (2). To this end, the auxiliary Algorithm 2 produces a list of models with top R^2 values by looping through all candidate models of size r. These models are returned as an index matrix $\boldsymbol{M}$, indicating selected terms with a 1 and deselected terms with a 0, where each column stands for a candidate model (lines 5,11). The models are sorted in descending order with respect to R^2 (line 13).

Back to Algorithm 3, we successively increase r starting from $r = 1$ (lines 7,22). We found that for a fixed r value, R^2 performs particularly well in identifying the best model out of the millions of models (see for instance Figure **??**). To infer a value for r with just R^2, we create a feature rating matrix $\boldsymbol{F}$ defined as the weighted counts of terms being selected across s top models, where the weight is given by R^2. Based on $\boldsymbol{F}$, we check for terms selected by R^2 for two successive model sizes r and $r-1$ (lines 16-20). If for both r and $r-1$ the same terms are selected consistently, we choose these two terms as part of the inferred model and conclude the search.

As for larger values of r the number of candidate models can easily reach hundreds of millions, we implement another strategy to divide out the list of candidate models. If terms have not been selected for two successive model sizes r and $r-1$, we remove these terms from the design matrix $\boldsymbol{K}$ (lines 14-15). In this way, we continuously reduce the model equation space as we go along.

The rationale for this selection and elimination strategy being solely based on R^2 is the following: If the true model has r terms, and we are testing all models with $r-2$ terms, then the models with largest R^2 will consistently be composed of the $r-2$ terms that contribute the most to explaining the variance of y. The other 2 true terms will be selected sporadically but at least once, terms that have not been selected at all can hence be removed from the candidate models. Testing in the next stage all models with $r-1$ terms, one more term will be consistently selected. The same holds for testing models with r terms, but when testing models with $r+1$ terms, no new term can contribute consistently to explaining more of the variance of y. The R^2 measure will increase for models for $r+1$ terms, but compared to models with r terms, no extra term will consistently be selected. Therefore, once no new term is selected consistently when incrementing the number r of terms, we can conclude that all contributing terms have been found. An illustration of this strategy can be found in Figures **??** and **??** in appendix **??**, where a typical case is shown.

Since the cheap computation of R^2 allows to go through millions of models in a matter of minutes on a standard computer, together with the described SCS strategy, we are able to consider or exclude all candidate models that can be built from the basis function dictionary.

In a final step, the list of top models from the R^2 evaluation can be combined and each tested with other selection criteria like $p(y|\mathcal{M})$, AIC, BIC or R^2_{adj}.

Algorithm 3 Semi-comprehensive search with R^2

Function SC-SEARCH($\boldsymbol{y}, \boldsymbol{X}$)
Input: data $\boldsymbol{y}, \boldsymbol{X}$
Parameters: maximum number $r_{\max}$ of terms, number s of top models for feature selection
$\boldsymbol{K}, p :=$ DESIGNMATRIX($\boldsymbol{X}$)
initialize feature rating $\boldsymbol{F}$ of shape $p \times r_{\max}$ # rate importance of terms for different model sizes r
initialize $search := True$
initialize number of terms $r := 1$
while $search$ and $r \leq r_{\max}$ **do**
  $\boldsymbol{M}, \boldsymbol{c} :=$ TopRsq($\boldsymbol{y}, \boldsymbol{K}, r$) # obtain list of models and with their R^2 values for fixed model size r
  append $\boldsymbol{M}$ to $\boldsymbol{M}_{\text{all}}$ # append models to index matrix keeping models for all r
  $\boldsymbol{F}_{:,r} := \sum_j^s c_j \, \boldsymbol{M}_{:,j}$ # counts how often terms are selected in s top models, weighted by R^2 criterion
  normalise $\boldsymbol{F}_{:,r} := \boldsymbol{F}_{:,r} \,/\, \max(\boldsymbol{F}_{:,r})$ # to have ratings between 0 and 1
  if $r \geq 2$ **then**
    index $\boldsymbol{i}_0 := (\boldsymbol{F}_{:,r} + \boldsymbol{F}_{:,r-1} = 0)$ # i_0=True if terms not selected for two successive model sizes
    remove $\boldsymbol{K}[:, \boldsymbol{i}_0]$ from $\boldsymbol{K}$ # reduce model equation space by those terms
    index $\boldsymbol{i}_1 := (\boldsymbol{F}_{:,r} \geq 0.75)$ # indexes terms with significant rating across best s models of size r
    index $\boldsymbol{i}_2 := (\boldsymbol{F}_{:,r-1} \geq 0.75)$ # same for previously conidered model size $r-1$
    if $\boldsymbol{i}_1 = \boldsymbol{i}_2$ **then**
      $search := False$ # if term is selected twice in a row like this, conclude search
    end if
  end if
  $r := r + 1$
end while
Compute criteria $\{\, p(y|\mathcal{M}), \text{AIC}, \text{BIC}, R^2_{\text{adj}} \,\}$ for $\boldsymbol{M}_{\text{all}}$
Return: $\boldsymbol{M}_{\text{all}}$ along with criteria

5 RESULTS AND DISCUSSION

5.1 RANDOM POLYNOMIALS

To put our strategy to the test, we randomly generated 100 polynomials with 2 and 3 non-zero weights respectively. We restricted the terms of the polynomials to have a maximum power $M_2 = 4$, where individual features are restricted to maximum power $M_1 = 2$. The coefficients w_n of the polynomials are sampled uniformly from the interval $\pm[1, 4]$ where the sign is picked at random with equal probability. To generate observations from each polynomial, we randomly generated N datapoints $\boldsymbol{x}_i$ with three features (x_{i1}, x_{i2}, x_{i3}), sampled uniformly from three random intervals nested in $[-10, 10]$, respectively. Plugging x_{ij} into (1) with $f(\boldsymbol{x}_i)$ given by the random polynomial, and corrupting the output with normal noise with standard deviation $\sigma = 0.01$, we generate data y_i for the response variable.

Using the artificially generated data $(\boldsymbol{x}_i, y_i)$, we employ Algorithm 3 to uncover the underlying true polynomials. We cut feature power at $M_1 = 4$ and term power at $M_2 = 6$, leading to a total of $p = 71$ terms.The identification rates for polynomials are shown Figure 1.

5.2 PREDICTIONS ON DYNAMICAL SYSTEMS

Next we apply the equation learning strategy to dynamical systems as explained in Section 3. We use the same hyper-parameters as for uncovering random polynomials in the previous section 5.1.

As a first non-chaotic example, we use an epidemiological model taken from Schlickeiser & Kröger (2021). This model has 4 compartments, *Susceptible*, *Infected*, *Recovered*, *Vaccinated*. The proportions of a population being in these self-descriptive categories are denoted by S, I, R, and V, respectively, hence the abbreviation SIRV for this model. The time evolution equation are of the form

$$\dot{S}(t) = -S(t)\,\big(a\,I(t) + v\big), \quad \dot{I}(t) = I(t)\,\big(a\,S(t) - \mu\big), \quad \dot{R}(t) = \mu I(t), \quad \dot{V}(t) = vS(t). \tag{15}$$

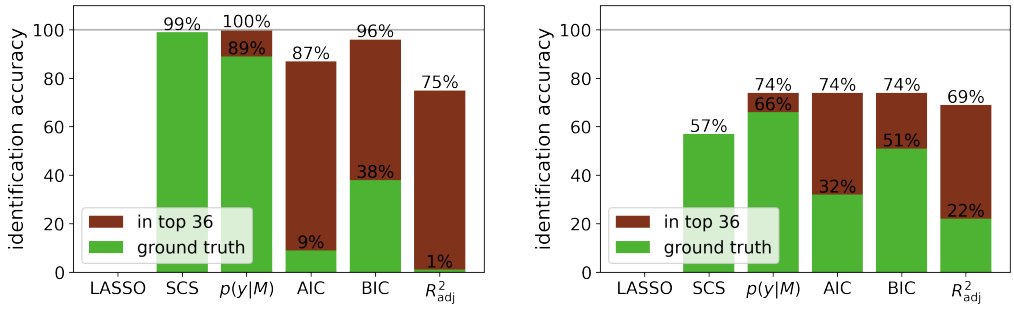

Figure 1: Success rates of Algorithm 3 for uncovering true polynomials with maximum number $r_{\max} = 8$ of terms. In this simulation, we used 100 random polynomials *left panel*: with 2 terms, from which we generated $N = 20$ datapoints; *right panel*: with 3 terms, from which we generated $N = 60$ datapoints

We fix the vaccination rate to $v = 0.05$, the recovery rate to $\mu = 1$, and the infection rate to $a = 3$. Using 4 similar initial conditions,

we solve (15) with a standard Runge-Kutta scheme (RK45) from SCIPY.INTEGRATE for a total time of $T = 1$ and $T = 2$, obtaining $N = 100$ datapoints for each initial value, respectively. We derive the response variable y from (14) and corrupt it with normal noise with standard deviation $\sigma = 0.002$.

Using the data for $T = 1$ and $T = 2$, we use Algorithm 3 with the model evidence $p(y|\mathcal{M})$ as final selection criterion. For $T = 1$, we are recovering all terms apart from the first term for $S(t)$, and for $T = 2$ we are completely recovering the structure of (15). Determining the coefficients using (3), we solve the inferred models numerically and compare it in Figure 2 against the true solution up to $t = 10$.

While the SIRV model was used to illustrate the prediction accuracy of our method in the case of limited data on a merely qualitative level, we now turn to an in-depth analysis of our second example, the chaotic Lorenz system,

$$\dot{x}(t) = \epsilon\big(y(t) - x(t)\big)\ , \quad \dot{y}(t) = x\big(\rho - z(t)\big) - y\ , \quad \dot{z}(t) = x(t)y(t) - \beta z(t). \tag{16}$$

The parameters are fixed to its standard values $\epsilon = 10, \rho = 28, \beta = 8/3$. As initial values we use $(x = -20, y = 20, z = 2)$ to obtain data for training by solving the system numerically with the Runge-Kutta scheme.

For the Lorenz system as a typical benchmark system for dynamical system identification, we perform a comparison to SINDy as a state-of-the-art technique Brunton et al. (2016). To this end, we consider 60 different scenarios generating different training datasets, in which the size of datasets range from $N = 200$ to $N = 10000$, the training time T various between $T = 5$ and $T = 50$, and the noise magnitude σ takes values between $\sigma = 0.0001$ and $\sigma = 0.2$. For each of these scenarios, we learn the model equations from the generated data using our Algorithm 2, which includes the LASSO, SCS, evidence $p(y, \mathcal{M})$, BIC, AIC, and R^2_{adj}.

In addition, also the python package PYSINDY de Silva et al. (2020) was used to perform dynamical system identification with SINDy. For SINDy, we use the same settings as used for the Lorenz system example in the package documentation, in which the `STLSQ` optimizer (sequentially thresholded least squares algorithm) is selected. Testing a few other optimizers (c.f. SSR, SR3, FROLS) we found that STLSQ indeed works best in our numerical experiment. The maximum degree for the polynomials was set to 4 for SINDy, which is equivalent to setting $M_2 = 4$ in our Algorithm 1; controlling the individual powers of the features in the polynomials (fixed by M_1 in Algorithm 1) is not possible in PYSINDY.

To not only test the ability to correctly learn the model equations from the generated data, but also assess the prediction accuracy, we solved for each scenario the learnt model for 100 different random initial values numerically using LSODA, and computed the mean absolute error (MAE) between the solution of the learnt model and the solution of the correct model. The LSODA

solver is a Adams/BDF method with automatic stiffness detection and switching implemented in SCIPY.INTEGRATE and the preferred method in PYSINDY. The initial values were uniformly sampled from the interval $[-20, 20]$ for x, from $[-25, 25]$ for y, and from $[0, 50]$ for z. The integration time was fixed to $T = 15$ to allow convergence to the strange attractor of the Lorenz system. With the 60 scenarios, 7 equation learning methods, and 100 initial values each we have a total of $42,000$ predicted trajectories to base our analysis on. We believe that in this way we can provide an in-depth comparison between the various equation learning methods.

Given the large number of learnt models and initial values, some initial value problems are not solvable by LSODA. In these cases, less trajectories contributed to the MAE and we kept track of the number of failures for each method as reported in the results.

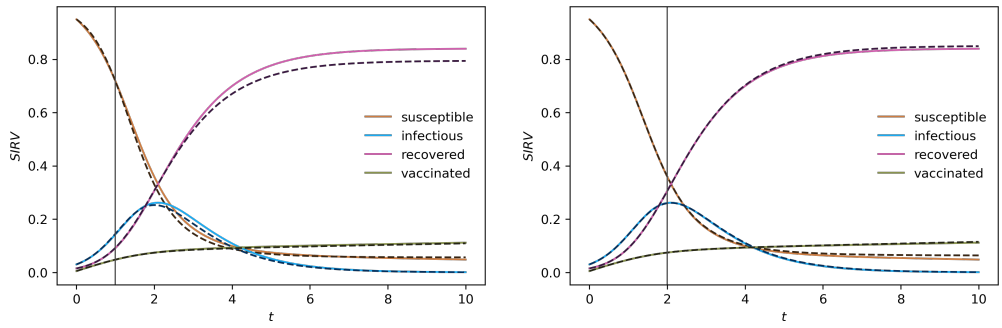

Figure 2: Predictions based on the model learnt from $N = 400$ noisy datapoints in the time interval $[0, 1]$ (left) and $[0, 2]$ (right). The noise level was $\sigma = 0.002$. The true time evolution is given by solid lines, the predictions by dashed lines. The ground truth is the SIRV model given by (15).

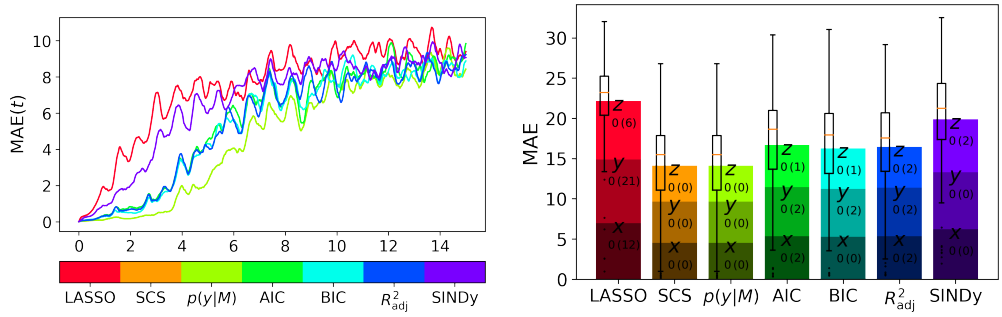

Figure 3: Comparison of prediction accuracy between various the equation learning methods. The left panel shows the evolution of the MAE across 60 scenarios with 100 initial values for the dynamical system learnt in each scenario. The right panel depicts the overal MAE, broken down into the x-, y- and z-component.

The results of the numerical experiments are summarised in 1. The MAE used for assessing the prediction accuracy is illustrated in **??** for one example.

5.3 DISCUSSION

We were able to uncover most of the polynomials. We stress the difficulty of this task, which consists in various aspects: i) Randomly generated polynomials are not hand-picked examples where some fine-tuning is possible. ii) The artificially generated data was not tested to represent the polynomial uniquely. iii) Even if the inferred polynomial is wrong by only one term and would still describe the data very well, it does not count as success in the final results. With these aspects in mind, it is quite remarkable that overall most of the polynomials could be recovered. Particular striking is the success rate of the semi-comprehensive search (SCS) strategy solely based on R^2. As mentioned before, the LASSO does not turn out to be suitable for uncovering the ground truth equations.

Table 1: Comparison of prediction accuracy: "all true terms" indicates how often the model includes all true terms, while "exactly true" requires that the exact Lorenz system has been recovered without extra terms. The MAE averaged over all timesteps, components, scenarios and initial values is shown under "MAE overall average", and "ranking in terms of MAE" indicates the rank (1-st to 7-th) of the methods in terms of the MAE averaged across scenarios and initial values. Lastly, "learnt model not solvable" reports how often an initial value caused a failure of LSODA. The first block uses all scenarios, the second block is restricted to $N \leq 1000$ datapoints.

	LASSO	SCS	$p(y\|\mathcal{M})$	AIC	BIC	R^2_{adj}	SINDy
all scenarios							
all true terms	**94.9 %**	67.8 %	74.6 %	79.7 %	76.3 %	79.7 %	78.0 %
exact true model	0.0 %	**66.1 %**	30.5 %	15.3 %	23.7 %	11.9 %	42.4 %
MAE overall average	9.02	6.49	6.21	**6.15**	6.22	6.19	7.03
ranking in terms of MAE	5.85	3.0	2.75	2.63	**2.39**	2.83	4.93
learnt model not solvable	20.66 %	0.69 %	**0.0 %**	0.15 %	0.15 %	0.15 %	6.34 %
scarce data ($N < 1000$)							
all true terms	**100.0 %**	54.1 %	73.0 %	73.0 %	73.0 %	73.0 %	67.6 %
exact true model	0.0 %	**51.4 %**	24.3 %	2.7 %	13.5 %	0.0 %	16.2 %
MAE overall average	10.28	7.14	**6.31**	6.45	6.39	6.5	7.95
ranking in terms of MAE	5.54	3.73	2.54	2.43	**2.03**	2.59	6.08
learnt model not solvable	32.08 %	1.11 %	**0.0 %**	0.24 %	0.24 %	0.24 %	10.11 %

The SIRV example demonstrated the ability to make accurate predictions to future times based on little data in case an underlying concise mathematical model exists that can be learnt. To also compare our method with SINDy as state-of-the-art, we used the well studied chaotic Lorenz system. The identification rate and the prediction accuracy turned out to be favourable compared to SINDy, in particular in the case of scarce data.

6 Conclusions

To our knowledge, we are the first to explore a deterministic model selection strategy that is based on an semi-comprehensive candidate model evaluation with the ability to compete with existing traditional and state-of-the-art methods. Being the first formulation of the method, it still holds plenty possibilities for improvements of the strategy.

A direct advantage is that the model evaluation is trivially parallelizable. Also the little amount of data needed for high success rates is striking – tests on all three sets of random polynomials where done with less datapoints than features in $\boldsymbol{K}$. Testing candidate models individually also allows for great flexibility when it comes to constraints or conditions on models, as well as eliminating the risk of getting stuck in local minima of an objective function. And finally, since our approach is fundamentally different from all approaches we are aware of, it can complement existing methods in an independent way with a potential of synergy effects.

While performance on the Lorenz system is substantially better than what has been achieved in the literature, our approach is still lacking the generality as has been achieved for example with SINDy Brunton et al. (2016) or neural networks Sahoo et al. (2018). Advancing these first steps, we expect our strategy to also be successful for more challenging examples systems involving fractions.

Considering this success and the surprisingly large class of models than can be described by linear regression models, we hope to open a new avenue of equation learning.

References

Forough Arabshahi, Sameer Singh, and Anima Anandkumar. Combining symbolic expressions and black-box function evaluations in neural programs. In *ICLR*, 2018.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, apr 2016. ISSN 0027-8424. doi: 10.1073/pnas.1517384113. URL http://www.pnas.org/lookup/doi/10.1073/pnas.1517384113.

Kathleen Champion, Peng Zheng, Aleksandr Y. Aravkin, Steven L. Brunton, and J. Nathan Kutz. A Unified Sparse Optimization Framework to Learn Parsimonious Physics-Informed Models From Data. *IEEE Access*, 8:169259–169271, 2020. ISSN 2169-3536. doi: 10.1109/ACCESS.2020.3023625. URL https://ieeexplore.ieee.org/document/9194760/.

Yize Chen, Marco Tulio Angulo, and Yang Yu Liu. Revealing Complex Ecological Dynamics via Symbolic Regression. *BioEssays*, 41(12):1–9, 2019. ISSN 15211878. doi: 10.1002/bies.201900069.

Alexandre Cortiella, Kwang Chun Park, and Alireza Doostan. Sparse identification of nonlinear dynamical systems via reweighted l1-regularized least squares. *Computer Methods in Applied Mechanics and Engineering*, 376:1–33, 2021. ISSN 00457825. doi: 10.1016/j.cma.2020.113620.

Brian de Silva, Kathleen Champion, Markus Quade, Jean-Christophe Loiseau, J. Kutz, and Steven Brunton. PySINDy: A Python package for the sparse identification of nonlinear dynamical systems from data. *Journal of Open Source Software*, 5(49):2104, 2020. ISSN 2475-9066. doi: 10.21105/joss.02104.

W.H. Donahue and O. Gingerich. *Johannes Kepler New Astronomy*. Cambridge University Press, 1992. ISBN 9780521301312. URL https://books.google.co.za/books?id=RCZpQgAACAAJ.

Renáta Dubčáková. Eureqa: software review. *Genetic Programming and Evolvable Machines*, 12(2):173–178, jun 2011. ISSN 1389-2576. doi: 10.1007/s10710-010-9124-z. URL http://link.springer.com/10.1007/s10710-010-9124-z.

Vincent Fortuin. Priors in Bayesian Deep Learning: A Review. (1):1–28, 2021. URL http://arxiv.org/abs/2105.06868.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York, New York, NY, 2nd edition, 2009. ISBN 978-0-387-84857-0. doi: 10.1007/978-0-387-84858-7. URL http://link.springer.com/10.1007/978-0-387-84858-7.

Marvin Höge, Thomas Wöhling, and Wolfgang Nowak. A Primer for Model Selection: The Decisive Role of Model Complexity. *Water Resources Research*, 54(3):1688–1715, mar 2018. ISSN 0043-1397. doi: 10.1002/2017WR021902. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/2017WR021902.

Ying Jin, Weilin Fu, Jian Kang, Jiadong Guo, and Jian Guo. Bayesian Symbolic Regression. 2019. URL http://arxiv.org/abs/1910.08892.

Kadierdan Kaheman, Steven L. Brunton, and J. Nathan Kutz. Automatic Differentiation to Simultaneously Identify Nonlinear Dynamics and Extract Noise Probability Distributions from Data. pp. 1–30, 2020. URL http://arxiv.org/abs/2009.08810.

Alan Kaptanoglu, Brian de Silva, Urban Fasel, Kadierdan Kaheman, Andy Goldschmidt, Jared Callaham, Charles Delahunt, Zachary Nicolaou, Kathleen Champion, Jean-Christophe Loiseau, J. Kutz, and Steven Brunton. PySINDy: A comprehensive Python package for robust sparse system identification. *Journal of Open Source Software*, 7(69):3994, 2022. doi: 10.21105/joss.03994.

Kevin H. Knuth, Michael Habeck, Nabin K. Malakar, Asim M. Mubeen, and Ben Placek. Bayesian evidence and model selection. *Digital Signal Processing: A Review Journal*, 47:50–67, 2015. ISSN 10512004. doi: 10.1016/j.dsp.2015.06.012. URL http://dx.doi.org/10.1016/j.dsp.2015.06.012.

Georg Martius and Christoph H. Lampert. Extrapolation and learning equations. 2016. URL http://arxiv.org/abs/1610.02995.

D. Montgomery, E. Peck, G. Vining, and an O'Reilly Media Company Safari. *Introduction to Linear Regression Analysis, 5th Edition*. John Wiley & Sons, 2012. ISBN 978-0-470-54281-1. URL https://books.google.co.za/books?id=hD46zQEACAAJ.

K.P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning series. MIT Press, 2012. ISBN 9780262018029. URL https://books.google.co.za/books?id=NZP6AQAAQBAJ.

John T. Nardini, John H. Lagergren, Andrea Hawkins-Daarud, Lee Curtin, Bethan Morris, Erica M. Rutter, Kristin R. Swanson, and Kevin B. Flores. Learning Equations from Biological Data with Limited Time Samples. *Bulletin of Mathematical Biology*, 82(9), 2020. ISSN 15229602. doi: 10.1007/s11538-020-00794-z.

R. Nayek, R. Fuentes, K. Worden, and E. J. Cross. On spike-and-slab priors for Bayesian equation discovery of nonlinear dynamical systems via sparse linear regression. *Mechanical Systems and Signal Processing*, 161:1–22, 2021. ISSN 10961216. doi: 10.1016/j.ymssp.2021.107986.

Pascal Neumann, Liwei Cao, Danilo Russo, Vassilios S. Vassiliadis, and Alexei A. Lapkin. A new formulation for symbolic regression to identify physico-chemical laws from experimental data. *Chemical Engineering Journal*, 387:123412, may 2020. ISSN 13858947. doi: 10.1016/j.cej.2019.123412. URL https://doi.org/10.1016/j.cej.2019.123412https://linkinghub.elsevier.com/retrieve/pii/S1385894719328256.

Robert K. Niven, Ali Mohammad-Djafari, Laurent Cordier, Markus Abel, and Markus Quade. Dynamical System Identification by Bayesian Inference. In *Proceedings of the 22nd Australasian Fluid Mechanics Conference AFMC2020*, dec 2020. doi: 10.14264/692fcb8. URL https://espace.library.uq.edu.au/view/UQ:692fcb8.

A. O'Hagan and M.G. Kendall. *Advanced Theory of Statistics: Bayesian inference. Volume 2B*. Number v. 2, pt. 2 in KENDALL, MAURICE GEORGE//KENDALL'S ADVANCED THEORY OF STATISTICS 6TH ED. Edward Arnold, 1994. ISBN 9780340529225. URL https://books.google.co.za/books?id=DlrEMgEACAAJ.

Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal Differential Equations for Scientific Machine Learning. 2020. ISSN 2331-8422. URL http://arxiv.org/abs/2001.04385.

Samuel H. Rudy, Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):1–7, apr 2017. ISSN 2375-2548. doi: 10.1126/sciadv.1602614. URL https://www.science.org/doi/10.1126/sciadv.1602614.

Subham S. Sahoo, Christoph H. Lantpert, and Georg Martius. Learning equations for extrapolation and control. *35th International Conference on Machine Learning, ICML 2018*, 10:7053–7061, 2018.

Hayden Schaeffer, Giang Tran, and Rachel Ward. Extracting Sparse High-Dimensional Dynamics from Limited Data. *SIAM Journal on Applied Mathematics*, 78(6):3279–3295, jan 2018. ISSN 0036-1399. doi: 10.1137/18M116798X. URL https://epubs.siam.org/doi/10.1137/18M116798X.

Reinhard Schlickeiser and Martin Kröger. Analytical modeling of the temporal evolution of epidemics outbreaks accounting for vaccinations. *Physics*, 3(2):386–426, 2021. ISSN 2624-8174. doi: 10.3390/physics3020028. URL https://www.mdpi.com/2624-8174/3/2/28.

David R. Stoutemyer. Can the Eureqa Symbolic Regression Program, Computer Algebra, and Numerical Analysis Help Each Other? *Notices of the American Mathematical Society*, 60(06):713, jan 2013. ISSN 0002-9920. doi: 10.1090/noti1000. URL http://www.ams.org/jourcgi/jour-getitem?pii=noti1000.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. ISSN 00359246. URL http://www.jstor.org/stable/2346178.

Robert Tibshirani and Jerome Friedman. A Pliable Lasso. *Journal of Computational and Graphical Statistics*, 29(1):215–225, 2020. ISSN 15372715. doi: 10.1080/10618600.2019.1648271.

Silviu Marian Udrescu and Max Tegmark. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16), 2020. ISSN 23752548. doi: 10.1126/sciadv.aay2631.

Harsha Vaddireddy, Adil Rasheed, Anne E. Staples, and Omer San. Feature engineering and symbolic regression methods for detecting hidden physics from sparse sensor observation data. *Physics of Fluids*, 32(1):015113, jan 2020. ISSN 1070-6631. doi: 10.1063/1.5136351. URL http://aip.scitation.org/doi/10.1063/1.5136351.

W Von Der Linden, R Preuss, and V Dose. The prior-predictive value: A paradigm of nasty multi-dimensional integrals. In *Maximum Entropy and Bayesian Methods Garching, Germany 1998*, pp. 319–326. Springer, 1999. URL https://link.springer.com/chapter/10.1007/978-94-011-4710-1_31.

Matthias Werner, Andrej Junginger, Philipp Hennig, and Georg Martius. Informed Equation Learning. pp. 1–24, may 2021. URL http://arxiv.org/abs/2105.06331.

Liu Yang, Xuhui Meng, and George Em Karniadakis. B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. *Journal of Computational Physics*, 425:109913, jan 2021. ISSN 00219991. doi: 10.1016/j.jcp.2020.109913. URL https://linkinghub.elsevier.com/retrieve/pii/S0021999120306872.

Sheng Zhang and Guang Lin. Robust data-driven discovery of governing physical laws with error bars. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2217):20180305, sep 2018. ISSN 1364-5021. doi: 10.1098/rspa.2018.0305. URL https://royalsocietypublishing.org/doi/10.1098/rspa.2018.0305.

Sheng Zhang and Guang Lin. SubTSBR to tackle high noise and outliers for data-driven discovery of differential equations. *Journal of Computational Physics*, 428:1–32, 2021. ISSN 10902716. doi: 10.1016/j.jcp.2020.109962.

Peng Zheng, Travis Askham, Steven L. Brunton, J. Nathan Kutz, and Aleksandr Y. Aravkin. A Unified Framework for Sparse Relaxed Regularized Regression: SR3. *IEEE Access*, 7: 1404–1423, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2018.2886528. URL https://ieeexplore.ieee.org/document/8573778/.

A Appendix

B EXACT BAYESIAN MODEL EVIDENCE USED FOR EQUATION LEARNING

The model is given by

$$y_i = \sum_n w_n \boldsymbol{K}_{in} + z_i/\sqrt{\tau} \tag{17}$$

where $\boldsymbol{K}_{in} = \boldsymbol{K}_n(x_i)$ is the basis function design matrix, w_n are the weights, $\tau = 1/\sigma^2$ is the precision, and $z \sim \mathcal{N}(0,1)$. For the whole vector $\boldsymbol{y}$ of N responses, we can use the multivariate normal for the likelihood,

$$p(\boldsymbol{y} \,|\, \boldsymbol{K}, \boldsymbol{w}, \tau) = \frac{\tau^{N/2}}{(2\pi)^{N/2}} \, \exp\Big(-\frac{\tau}{2} \, (\boldsymbol{y} - \boldsymbol{K}\boldsymbol{w})^{\mathrm{T}} \, (\boldsymbol{y} - \boldsymbol{K}\boldsymbol{w}) \Big), \tag{18}$$

where the precision matrix is diagonal with identical τ on the diagonal. Since the weight parameters w_n enter quadratically, we can rewrite this expression in normal form for $\boldsymbol{w}$,

$$p(\boldsymbol{y}|\boldsymbol{K}, \boldsymbol{w}, \tau) = \frac{\tau^{N/2}}{(2\pi)^{N/2}} \, \exp\Big(-\frac{\tau}{2} \, S \Big) \, \exp\Big(-\frac{\tau}{2} \, (\boldsymbol{w} - \hat{\boldsymbol{w}})^{\mathrm{T}} \, \boldsymbol{K}^{\mathrm{T}} \boldsymbol{K} \, (\boldsymbol{w} - \hat{\boldsymbol{w}}) \Big) \tag{19}$$

with the residual sum of squares

$$\begin{aligned} S &= (\boldsymbol{y} - \boldsymbol{K}\hat{\boldsymbol{w}})^{\mathrm{T}} (\boldsymbol{y} - \boldsymbol{K}\hat{\boldsymbol{w}}) && (20)\\ &= \boldsymbol{y}^{\mathrm{T}}\boldsymbol{y} - \hat{\boldsymbol{w}}^{\mathrm{T}} \boldsymbol{K}^{\mathrm{T}} \boldsymbol{y} && (21)\\ &= \boldsymbol{y}^{\mathrm{T}}\boldsymbol{y} - y^{\mathrm{T}} \boldsymbol{K} (\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K})^{-1} \boldsymbol{K}^{\mathrm{T}} \boldsymbol{y} && (22) \end{aligned}$$

The mixed terms cancel after plugging in $\boldsymbol{K}^{\mathrm{T}}\boldsymbol{y} = \boldsymbol{K}^{\mathrm{T}}\boldsymbol{K}\hat{\boldsymbol{w}}$ from the known OLS solution $\hat{\boldsymbol{w}} = (\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K})^{-1}\boldsymbol{K}^{\mathrm{T}}\boldsymbol{y}$.

The above is of the form of a gamma distribution for τ multiplied with a normal distribution for $\boldsymbol{w}$ conditioned on τ. If we use a prior of the same form, we keep the form for the posterior, and thus have found the conjugate prior.

As a prior for the weights $\boldsymbol{w}$, we choose

$$p(\boldsymbol{w} \,|\, \boldsymbol{\mu}, \boldsymbol{M}) = \frac{\tau^{p/2}\sqrt{\det \boldsymbol{M}}}{(2\pi)^{p/2}} \, \exp\Big(-\frac{\tau}{2} \, (\boldsymbol{w} - \boldsymbol{\mu})^{\mathrm{T}} \, \boldsymbol{M} \, (\boldsymbol{w} - \boldsymbol{\mu}) \Big), \tag{23}$$

where $\tau\boldsymbol{M}$ is the precision matrix with τ split off, and $\boldsymbol{\mu}$ is the mean vector of the multivariate normal prior. Splitting off τ technical means that specifying $\boldsymbol{M}$ is relative to the unknown τ, but τ does not need to be known for that, as we integrate over all possible τ values.

For the posterior, we are interested in the quadratic form involving $\boldsymbol{w}$,

$$(\boldsymbol{w} - \hat{\boldsymbol{w}})^{\mathrm{T}} \, \boldsymbol{K}^{\mathrm{T}} \boldsymbol{K} \, (\boldsymbol{w} - \hat{\boldsymbol{w}}) + (\boldsymbol{w} - \boldsymbol{\mu})^{\mathrm{T}} \, \boldsymbol{M} \, (\boldsymbol{w} - \boldsymbol{\mu}) = \boldsymbol{w}^{\mathrm{T}} \, \boldsymbol{A} \, \boldsymbol{w} - 2 \, \boldsymbol{w}^{\mathrm{T}} \, \boldsymbol{b} + c \tag{24}$$

with

$$\begin{aligned} \boldsymbol{A} &= \boldsymbol{K}^{\mathrm{T}}\boldsymbol{K} + \boldsymbol{M} && (25)\\ \boldsymbol{b} &= \boldsymbol{K}^{\mathrm{T}}\boldsymbol{K}\hat{\boldsymbol{w}} + \boldsymbol{M}\boldsymbol{\mu} \\ &= \boldsymbol{K}^{\mathrm{T}}\boldsymbol{y} + \boldsymbol{M}\boldsymbol{\mu} && (26)\\ c &= \hat{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K}\hat{\boldsymbol{w}} + \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{\mu} \\ &= \hat{\boldsymbol{w}}^{\mathrm{T}}\boldsymbol{K}^{\mathrm{T}}\boldsymbol{y} + \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{\mu} && (27)\\ &= y^{\mathrm{T}}\boldsymbol{K}(\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K})^{-1}\boldsymbol{K}^{\mathrm{T}}\boldsymbol{y} + \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{\mu}. && (28) \end{aligned}$$

Put into this form, we can use the Gaussian integral $\int \mathrm{d}^n x\, e^{-\frac{1}{2}x^{\mathrm{T}}\boldsymbol{A}x+b^{\mathrm{T}}x} = \sqrt{\frac{(2\pi)^n}{\det \boldsymbol{A}}}\, e^{\frac{1}{2}b^{\mathrm{T}}\boldsymbol{A}^{-1}b}$ to marginalise,

$$
\begin{aligned}
p(\boldsymbol{y}\,|\,\tau) &= p(\boldsymbol{y}\,|\,\boldsymbol{K},\tau,\boldsymbol{\mu},\boldsymbol{M}) && (29)\\
&= \int \mathrm{d}\boldsymbol{w}\; p(\boldsymbol{y}\,|\,\boldsymbol{K},\boldsymbol{w},\tau)\, p(\boldsymbol{w}\,|\,\boldsymbol{\mu},\boldsymbol{M}) && (30)\\
&= \frac{\sqrt{\tau^{N+p}\det \boldsymbol{M}}}{\sqrt{(2\pi)^{N+p}}}\, e^{-\frac{\tau}{2}(S+c)} \int \mathrm{d}\boldsymbol{w}\, e^{-\frac{\tau}{2}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{w}-2\boldsymbol{b}^{\mathrm{T}}\boldsymbol{w})} && (31)\\
&= \frac{\sqrt{\tau^{N+p}\det \boldsymbol{M}}}{\sqrt{(2\pi)^{N+p}}}\, e^{-\frac{\tau}{2}(S+c)}\, \frac{\sqrt{(2\pi)^p}}{\sqrt{\tau^p \det \boldsymbol{A}}}\, e^{\frac{\tau}{2}\boldsymbol{b}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{b}} && (32)\\
&= \sqrt{\frac{\tau^N \det \boldsymbol{M}}{(2\pi)^N \det \boldsymbol{A}}}\, e^{-\frac{\tau}{2}(\boldsymbol{y}^{\mathrm{T}}\boldsymbol{y}+\boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{\mu}-\boldsymbol{b}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{b})}. && (33)
\end{aligned}
$$

For the τ-integration, we choose the gamma distribution

$$
p(\tau|k,\vartheta) = \frac{1}{\Gamma(k)\vartheta^k}\, \tau^{k-1}\, \exp\big(-\tau/\vartheta\big) \tag{34}
$$

as the (conjugate) prior for τ, and define

$$
\xi = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{y} + \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{M}\boldsymbol{\mu} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{b}. \tag{35}
$$

The remaining τ-integral follows then from $\int_0^\infty \mathrm{d}\tau\, \tau^{c_0} e^{-c_1\tau} = c_1^{-c_0-1}\,\Gamma(c_0{+}1)$ as

$$
\begin{aligned}
p(y) &= p(y\,|\,\boldsymbol{K},\boldsymbol{\mu},\boldsymbol{M},k,\vartheta) && (36)\\
&= \int_0^\infty \mathrm{d}\tau\, p(\tau|k,\vartheta)\, p(\boldsymbol{y}\,|\,\tau) && (37)\\
&= \frac{1}{\Gamma(k)\vartheta^k(2\pi)^{\frac{N}{2}}} \sqrt{\frac{\det \boldsymbol{M}}{\det \boldsymbol{A}}} \int_0^\infty \mathrm{d}\tau\, \tau^{\frac{N}{2}+k-1}\, e^{-\tau(\frac{z}{2}+\frac{1}{\vartheta})} && (38)\\
&= \frac{\Gamma(\frac{N}{2}+k)}{\Gamma(k)\vartheta^k(2\pi)^{\frac{N}{2}}} \sqrt{\frac{\det \boldsymbol{M}}{\det \boldsymbol{A}}} \left(\frac{z}{2}+\frac{1}{\vartheta}\right)^{-\frac{N}{2}-k} && (39)
\end{aligned}
$$

and for the log-evidence per data-point we obtain

$$
\begin{aligned}
\frac{1}{N}\ln p(\boldsymbol{y}) = \frac{1}{2N}\ln\frac{\det \boldsymbol{M}}{\det \boldsymbol{A}} - \frac{1}{2}\ln 2\pi - \Big(\frac{1}{2}+\frac{k}{N}\Big)\ln\Big(\frac{\xi}{2}+\frac{1}{\vartheta}\Big)\\
- \frac{k}{N}\ln\vartheta + \frac{1}{N}\ln\Gamma\Big(\frac{N}{2}+k\Big) - \frac{1}{N}\ln\Gamma(k). \qquad (40)
\end{aligned}
$$