

Physics without Laws - Making exact Predictions with data based Methods

Lars Kindermann

Lab for Mathematical Neuroscience
RIKEN Brain Science Institute, Japan
kindermann@brain.riken.go.jp
www.reglos.de

Peter Protzel

Dept. of Electrical Engineering and Information Technology
Chemnitz University of Technology, Germany
peter.protzel@e-technik.tu-chemnitz.de
www.infotech.tu-chemnitz.de/~proaut

Abstract - The mathematical method of fractional or continuous iteration can be used to model a dynamical system exactly from limited experimental data. However, mathematics is complicated and exact solutions - even if proven to exist - can rarely be found analytically. We have shown previously that neural networks can be utilized to numerically compute fractional iterates of mathematical functions. In this paper we demonstrate the application of this method to the fundamental experiment of physics: The free fall.

1 INTRODUCTION

The classical process of science can be viewed in a simplified way as a repetition of the scheme: ...make some experiment → make up a theory that fits the experimental data → make predictions → test them against experiment → modify theory... until theory and experiment agree.

Knowledge based methods try to avoid this stony road by feeding the available data into some algorithm, a process called (machine) learning, and use the generated model to make some predictions.

Data → Model → Predictions

The model is usually considered as a black box, you don't care about it's internal structure as long as it reproduces the experimental data correctly.

But the predictions are tightly bound to the same domain as the available data. Usually only changes of numerical values of the inputs are allowed. Any other question not matching the structure of the examples is usually beyond the capabilities of these models. This also means, that the model may not be applied to even slightly different problems. If something changes in the system that has created the training data, it is in general not possible to modify the generated model accordingly without retraining on new data.

Current trends in research tries to overcome these limitations by ideas based on rule extraction. In addition to generating the "black box" another goal is to extract human understandable rules, for example fuzzy rules. These rules are regarded as a more analytical description of the model than a weight matrix. "Understandable" is often used synonymously with "readable by humans".

We will use another approach here, a kind of structured

black box". Instead of using the fully connected, single hidden layer network which is known to be sufficient to approximate every given function, we propose to use topologically structured networks where substructures and their connections are supposed to have some "meaning". Mathematically such a network represents some functional equation. Parts of the whole network can be separated or rearranged later due to their „algebraic“ meaning to solve new questions.

To demonstrate this idea we will generate some training data from a simple experiment. But later we ask what will happen if we change a parameter of the setting which has been constant all the time before.

The "scientific" way of answering such questions is trying to find an equation describing the dynamics of this system, perhaps with free parameters which then are fitted to match the data. A system usually is considered "understood" if we find a differential equation describing its behavior. Integrating this equation may still be a serious mathematical problem, often only possible by numerical methods.

Once a theory is found, it can be used to make predictions and these predictions may span a much wider range than the previous experimental context. In fact, it is required by the "falsification principle" that a theory should be able to make predictions not evidently derivable from known experiments to be regarded as a "serious" theory.

In this paper we demonstrate how neural networks can be used to make calculations *as exact* as the training data for variations of the experiment not covered by the data set without the need for a mathematically formulated theory.

2 THE FREE FALL EXPERIMENT

A. The Experimental View

We will examine the very original problem which led to the foundation of modern physics in the Renaissance, the free fall of a body from a purely experimental view: Assume, we can make measurements only at two fixed points in space, e.g. at the top of a tower and on the ground or at the two endpoints of a tube of some fixed length.

Without being able to measure what happens to the object inside the tube, but with the freedom of choice to start with different speeds v_0 , we measure its speed v_1 when it leaves the

tube. What can one tell in principle about the speed at some point within the tube just from this data?

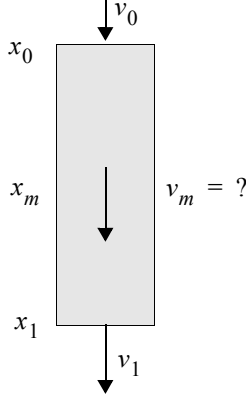


Figure 1: An object falls down inside a tube where it is impossible to make measurements. Can its speed be calculated by a neural network?

Instead of making a theory about the physics of free fall which might lead to the fundamental differential equation $\ddot{x} = g$ with the gravitational acceleration g , we will use a neural network to answer this question (almost) exactly (almost) without putting in any theory.

The experiment, if carried out with a representative range of different speeds v_0 provides us with a table $v_0 \rightarrow v_1$, defining the function $v_1 = f(v_0)$ by means of examples. Requesting the outcome of this experiment for some speed not yet determined by experiment is an interpolation problem (or extrapolation in case v_0 is selected bigger or smaller than in all experiments).

This kind of application is one of the main domains of neural networks. So let us assume we have created a sufficiently exact (neural) model for interpolating f .

As the function $f(v)$ describes the change of speed after passing the tube, $f(f(v))$ will give the speed after passing *two* tubes of the same kind, $f^n(v_0)$ after n tubes. When assuming f as bijective, f^{-1} will calculate the speed *before* passing through the tube.

The $f^n, n \in \mathbb{Z}$ form the so called *iteration group* of f with

$$f^{a+b}(v) = f^{b+a}(v) = f^b(f^a(v)) = f^a(f^b(v))$$

and $f^0(v) = v$ as the neutral element of this group. Embedding this into a *continuous* iteration group means finding solutions F of the *translation equation*

$$F(x_1 + x_2, v) = F(x_1, F(x_2, v))$$

for real valued x and v under the condition

$$F(n, v) = f^n(v)$$

for all integers n [7].

In case f is not invertible, only a semigroup can be constructed this way, but this won't affect our arguments here.

Setting $f^x(v) = F(x, v)$ by definition, the operation of iteration can formally be generalized to non-integer iteration counts this way.

Half an iteration of some function, $f^{1/2}(v)$ is usually called an *iterative root* of f and is a solution of the functional equation

$$\varphi(\varphi(v)) = f(v).$$

In our tube example $f^{1/2}(v_0)$ corresponds to the speed in the middle of the tube. Assuming without loss of generality $x_0 = 0$ and $x_1 = 1$, the speed at any location x would formally conform to

$$v(x) = f^x(v_0).$$

The proof of existence of such an embedding is difficult. At least it is known that every continuous monotonically rising function is embeddable [7].

B. Expected results by known laws

For the free fall we can verify this scheme easily: With acceleration g , applying conservation of energy law gives

$$\frac{1}{2}mv_0^2 + mg\Delta x = \frac{1}{2}mv_1^2$$

so the function f describing the observed data would look like

$$v_1 = f(v_0) = \sqrt{v_0^2 + 2g\Delta x} \quad (1)$$

With $\Delta x = 1$, the functional equation of type

$$\varphi(\varphi(v)) = \sqrt{v^2 + 2g}$$

has to be solved to get the speed in the middle of the tube. In this case a solution can easily be guessed:

$$\varphi(v_0) = \sqrt{v_0^2 + g}$$

Trivial proof:

$$\varphi(\varphi(v_0)) = \sqrt{(\sqrt{v_0^2 + g})^2 + g} = \sqrt{v_0^2 + 2g} = f(v_0)$$

But remember, just from the experiment we do not know f in algebraic form but only as a set of sample data which may be transformed to some (neural) model. If we provide an algorithm which can compute the iterative root of this implicit function, we could answer the question for the speed at v_m with the same precision as the measurements without requiring any explicit physical theory used here so far.

The only physical assumption made is the isotropy of space, expressed by the translation equation.

A continuous iterate of f that models the speed at any point x is of course also easy to guess here:

$$v(x) = f^x(v_0) = \sqrt{v_0^2 + 2ax} \quad (2)$$

Finding this relation numerically only from the experimental data would enable us to make precise predictions without dealing with any formalism.

Iterative roots of monotonically increasing continuous functions can be constructed by a method from Hardy [1]. But this also proves that there is, in general, no unique solution for the iterative root, solutions may depend on some arbitrary function. So in fact we are dealing with an “ill posed” problem here. This is probably one of the reasons why this theory remained almost unknown so far.

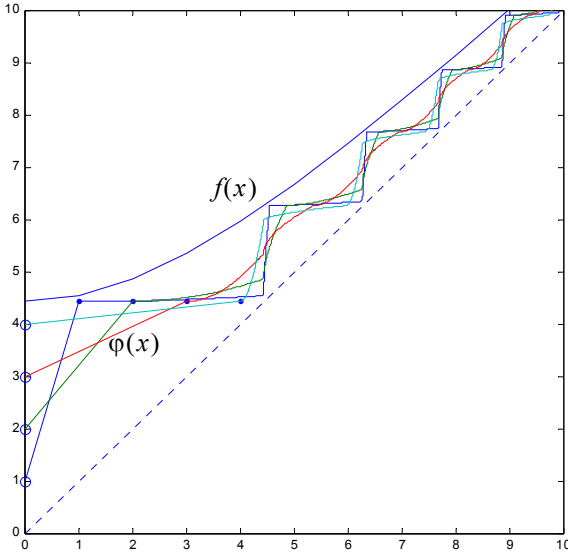


Figure 2: Construction of arbitrary solutions of the iterative root of eq. 1: Select an arbitrary value for $\varphi(0)$. Because $\varphi(\varphi(0)) = f(0)$ all values for the iterates $\varphi^n(0)$ are determined now as well. Connect the borders of the interval $[0, \varphi(0)]$ by some arbitrary continuously monotonically rising function now. This which will determine φ completely.

3 COMPUTING FRACTIONAL ITERATES WITH A NEURAL NETWORK.

Some straight forward extensions to the MLP model can be used to compute approximations of iterative roots and fractional iterates [8].

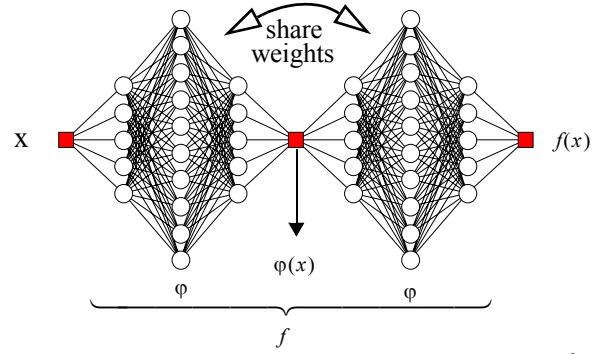


Figure 3: A MLP computes an iterative root of the function f

The basic idea is to use a structured network like Figure 3 to approximate f and additionally force the weights of the two subnets to be identical. Once both goals are reached, each subnet will be a model for the iterative root of f .

Different algorithms for training such networks were presented by us in previous papers [8, 10]

This method can be extended to n -th iterative roots, where a composition of n subnetworks is trained towards f . A fractional iterate $f^{m/n}$ can then be constructed by m of these subnetworks. Continuous iteration f^x can be approximated by selecting $\frac{m}{n} \sim x$.

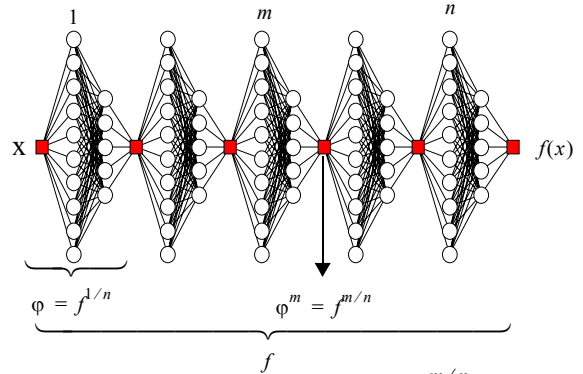


Figure 4: Finding a fractional iterate $f^{m/n}(x)$

Another method is to use a kind of recurrent network which is trained in n loops and later recalled in m loops [10].

4 THE FREE FALL WITH A NET

100 data samples were generated according to eq. 1. for $v_0 \in (0, 10(m/s))$. A MLP with topology 1-6-1 was able to approximate this data to an accuracy of 10^{-7} m/s (MSE). Then a network like shown figure 4 with an $(1-6-1)^4$ topology and weight sharing was used to compute the iterative root of this data, supposed to model the speed at x_m . The approximation error for the whole network remained the same. The result of one subnet, modeling the iterative root, was compared to the expected values from eq. 2.

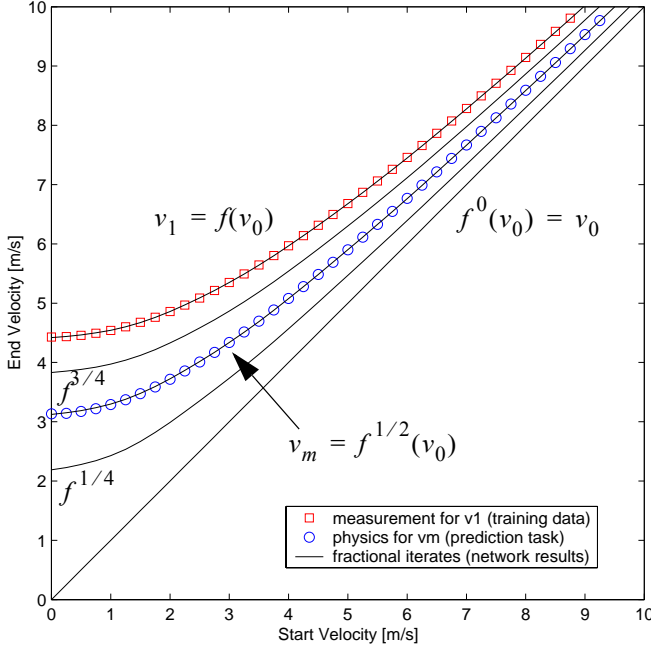


Figure 5: Iterative root of eq. 1 computed by a MLP, compared to the expected result from eq. 2. The speed in the middle of the tube, $f^{1/2}$, is reproduced with high accuracy now. Also shown are $f^0, f, f^{1/4}, f^{3/4}$.

Even though mathematics proves that there is no unique solution for the iterative root, the difference to the expected result was also only about 10^{-6} m/s.

Only when using more than about 100 neurons we were able to see some of the other arbitrary solutions, the “error“ for v_m started to grow. But 100 neurons also produced a similar error for v_1 on a cross validation set for f and would thus be disregarded anyway.

The recommended strategy is: First find the optimal and minimal network which approximates f . Then combine networks of the same size to compute the fractional iterates. This is suggested by the assumption that an iterative root of a function has the same complexity as the function itself. While this is not true in general (iterative roots of $f(x) = -x$ e.g. are quite nasty), it holds for most practical applications considered so far.

5 SOME FRICTION

While the presented example may seem not to be very useful as a real application, the fall with friction is far more complicated to deal with analytically. Suppose one has the same setting as before but we add two terms of friction, one which is dominant for slow velocity $\sim v$ and another which is dominant at higher velocities in air $\sim v^2$. The fall of the object is then governed by the differential equation

$$m\ddot{x} = mg + k_1\dot{x}^2 + k_2\dot{x} \quad (3)$$

Integrating this equation to get a closed formula describing

$v = f(x, v_0)$ is pretty difficult.

Moreover, if there is only experimental data available, describing input vs. output speed and we do not know some or all of the values m, g, k_1 and k_2 , it suddenly becomes a very hard mathematical task to get the speed at x_m .

We simulate a falling object with $m = 10g, g = 9.81(m/s^2), |k_1| = 1, |k_2| = 1$ by numerically integrating eq. 3 with $x(t_0) = 0$ and $\dot{x}(t_0) = v_0$ to get the speed after one meter v_1 (training data) and after half a meter v_m (task to predict). Then, the same network as before is trained on the one meter data (v_0 vs. v_1) and the speed v_m is retrieved as the “iterative root”.

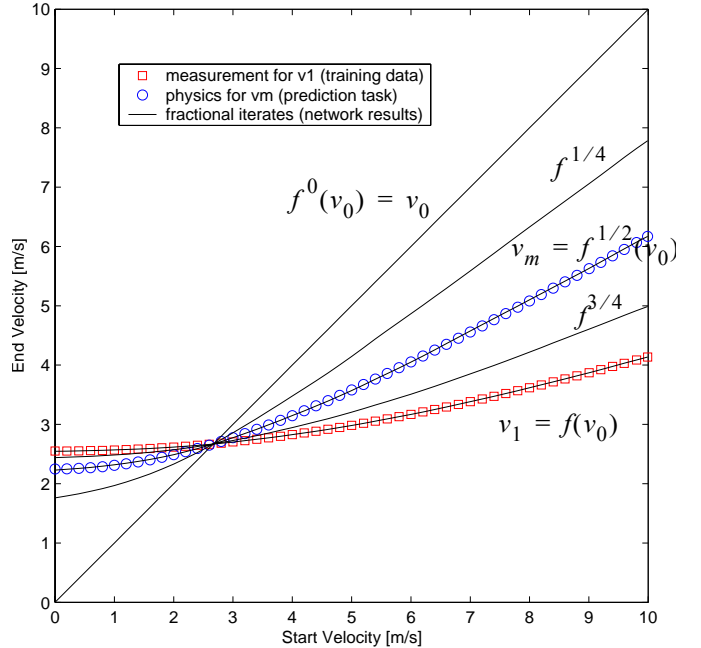


Figure 6: Modelling the free fall with friction

These results show that this nontrivial problem is solved with the same high accuracy, eliminating the need both to invent the model equation (3), solve it by integration and fit the unknown parameters to get an accurate model for v_m .

6 CONCLUSIONS

The surprising result of this example is: We can construct a neural network that models the physics of free fall - even with friction - from limited experimental knowledge almost perfectly. The network predictions for the speed at location x_m converge to that solution of the “iterative root” which conforms to physics up to about the same precision as the original function approximation.

The mathematical theory of embeddings, iterative roots and fractional iteration does suggest such an approach, but also proves that in general, it should be considered an “ill-posed“ problem.

Why among all the mathematically possible solutions of an embedding exactly the one which conforms to physics seems

to be selected is not yet clear. There exist attempts to assign uniqueness to iterative roots by adding constraints [11] but none have been implemented in our network model. Probably the restrictions of the MLP architecture with few neurons itself forces the solutions towards “simple” shapes which are also the constraints of physics.

Further research should be carried out to determine if more complex systems can also be modelled successfully by fractional iteration in general and the presented network approach for the practical solution. Mathematics still lacks an easy formalism to deal with this generalized iteration counts, which is not surprising at all as it is closely related to chaos theory - just the inverse direction than usual.

Our brain masters the physics of the free fall almost perfectly too - without solving differential equations analytically and with a limited amount of training data. This example proves there are structured networks which can use limited knowledge to generalize the complete dynamics of a simple systems. Perhaps we will find similar structures in the brain that encode basic properties of the physical world, in this case the translation invariance of the laws of mechanics, in a fixed network topology while fine tuning is necessary on the synaptic level (weight matrix) to accommodate for specific situations.

ONLINE

An extensive bibliography about iterative roots and fractional iteration as well as the MATLAB code for the neural networks used in this paper can be found at the authors website <http://www.reglos.de/kindermann/ffx.html>

REFERENCES

- [1] G.H. Hardy, E. Cunningham, *Orders of infinity*. Cambridge Tracts in Mathematics 12, 1924
- [2] M. Kuczma, B. Choczewski, R. Ger, *Iterative Functional Equations*. Cambridge University Press, Cambridge, 1990
- [3] K. Baron & W. Jarczyk, *Recent results on functional equations in a single variable, perspectives and open problems*. Aequationes Math. 61. (2001), 1-48
- [4] G. Targonski, *An Iteration theoretical approach to the concept of time*. Colloques Internationaux du C.N.R.S. 229, Transformations ponctuelles et leurs applications, Toulouse (1973), 259-271
- [5] G. Targonski, *Topics in iteration theory*. Vandenhoeck und Ruprecht, Göttingen 1981
- [6] G. Targonski, *Progress of iteration theory since 1981*. Aequationes Math. 50 (1995), 50-72
- [7] M.C. Zdun, *Continuous iteration semigroups*. Boll. Un. Mat. Ital. 14 A (1977), 65-70
- [8] L. Kindermann, *Computing Iterative Roots with Neural Networks*. Proceedings of the Fifth Conference on Neural Information Processing, ICONIP'98 Vol. 2:713-715, 1998
- [9] L. Kindermann, T.P. Trappenberg, *Modeling time-varying processes by unfolding the time domain*. Proceedings of the International Joint Conference on Neural Networks (IJCNN'99), Washington DC, 1999
- [10] L. Kindermann & A. Lewandowski, *A Comparison of Different Neural Methods for Solving Iterative Roots*. Proc. Seventh Int'l Conf. on Neural Information Processing - ICONIP'2000, Taejon (2000), 565-569
- [11] B.A. Reznick, *A uniqueness criterion for fractional iteration*. Ann. Polon. Math. 30 (1974), 219-224