

Physics without Math

Making exact Calculations with Data based Methods

Lars Kindermann



*RIKEN Brain Science Institute
Japan*

*Lab f. Mathematical Neuroscience
kindermann@brain.riken.go.jp
www.reglos.de/kindermann*

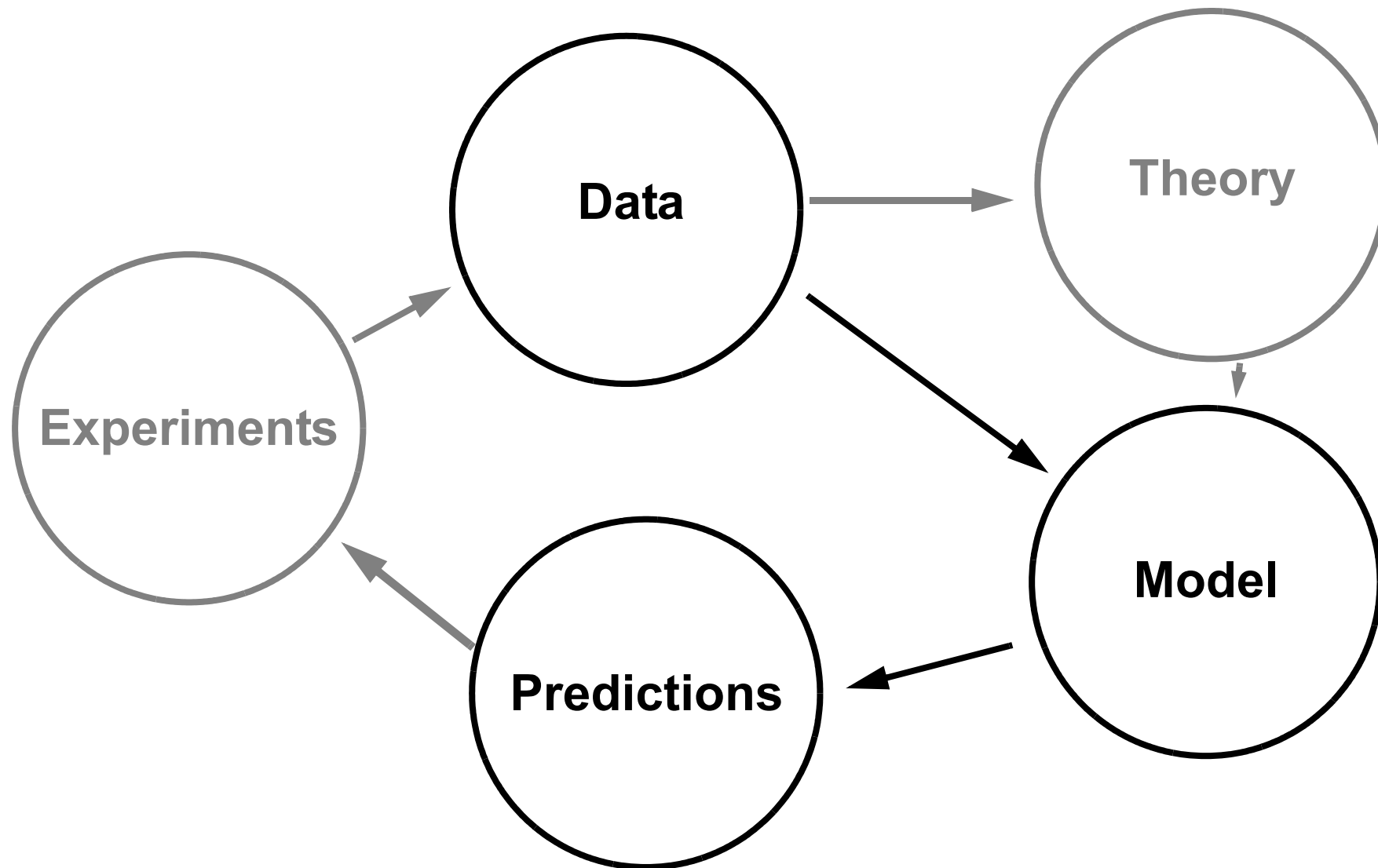
Peter Protzel



*Chemnitz University of Technology
Germany*

*Dept. of Electrical Eng. & Information Tech.
peter.protzel@e-technik.tu-chemnitz.de
www.infotech.tu-chemnitz.de/~proaut*

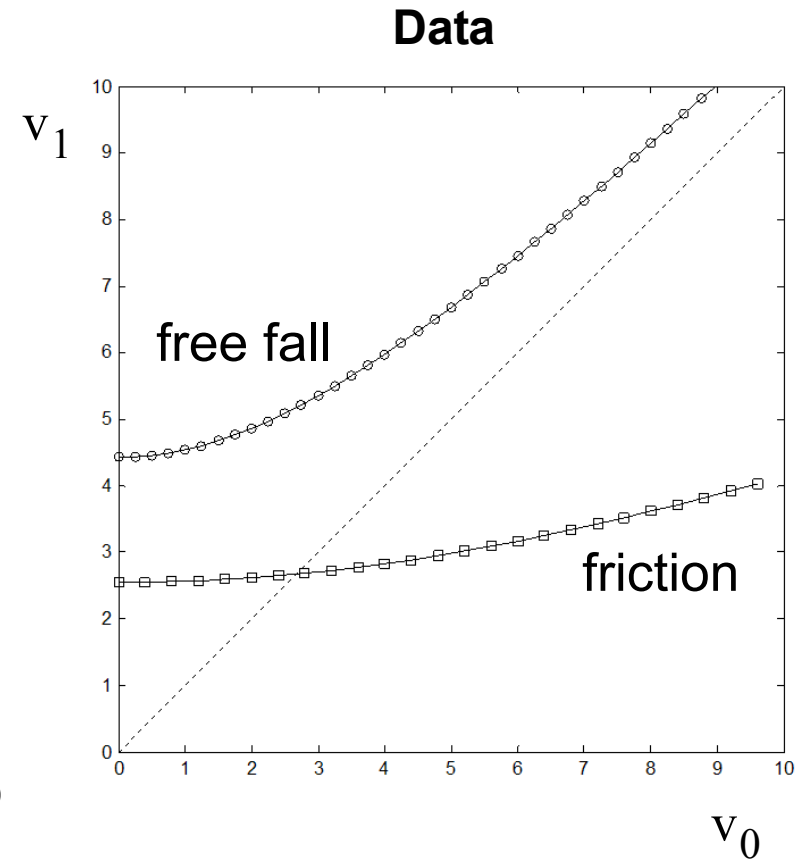
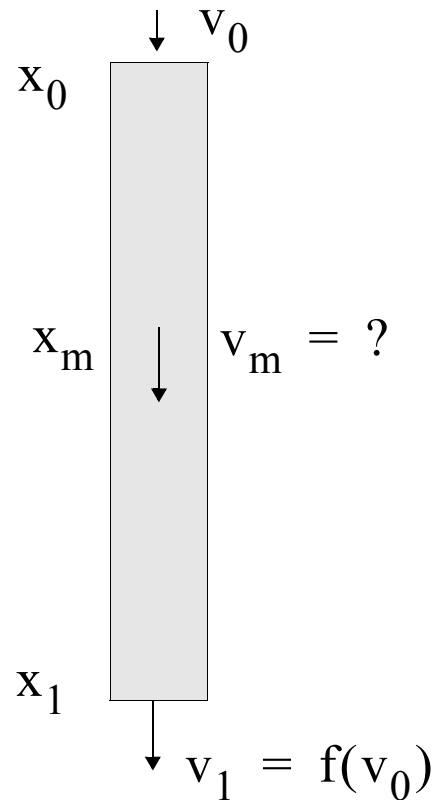
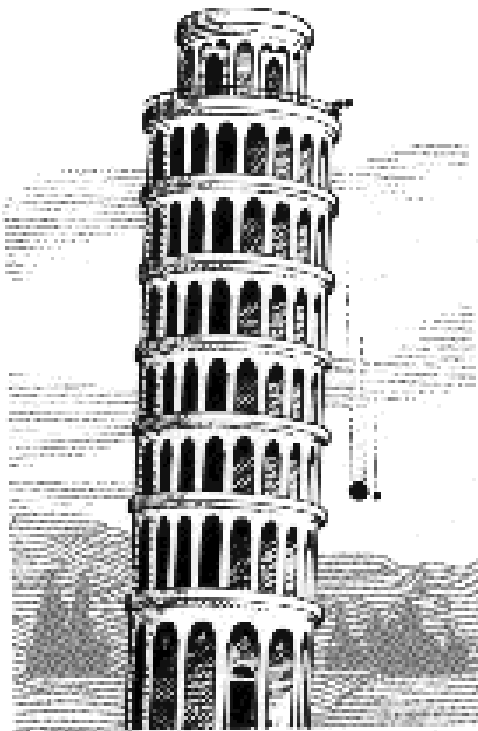
Science vs. Computational Intelligence



Theory: can answer many different questions - **Model:** valid for a specific Experiment only

An old Experiment revisited

Throw down an object with different speeds v_0 and measure the speed at the ground v_1



Question: What's the speed v_m after half the way at x_m ?

Solving it with Physics



Free Fall:

$$\text{Theory: } \frac{\partial^2 x}{\partial t^2} = g \Rightarrow$$

$$\text{Model: } v_1 = f(v_0) = \sqrt{v_0^2 + 2g\Delta x} \text{ with data fitted } g$$

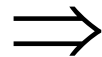
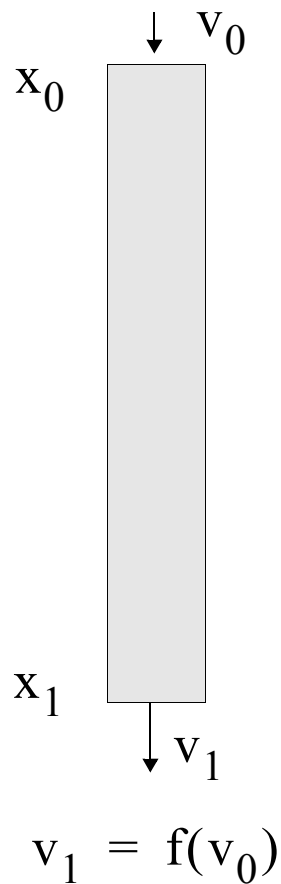
With additional Friction:

$$\text{Theory: } \frac{\partial^2 x}{\partial t^2} = g - k_1 \left| \frac{\partial x}{\partial t} \right| - k_2 \left| \frac{\partial x}{\partial t} \right|^2 - f\left(\frac{\partial x}{\partial t}\right) \Rightarrow$$

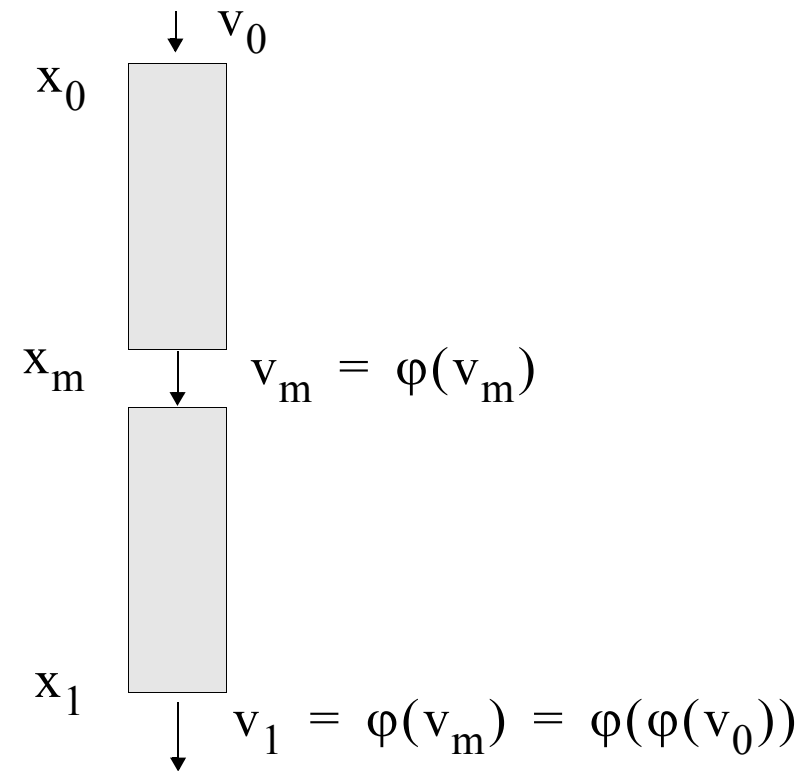
Model: Integrate numerically and fit g and k - already a non-trivial Problem!

Another Approach with less explicit Math

Theory: Assume translation invariance



divide into
two equal
steps...



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

and solve this functional equation for φ

Iterative Roots and Fractional Iteration



$$\varphi(\varphi(x)) = f(x)$$

A solution φ of this equation is called a *square root* of f .

- If $f(x): \mathbb{R}^n \rightarrow \mathbb{R}^n$ is a *function*, we look for another function $\varphi(x)$ which composed with itself equals f : $\varphi(\varphi(x)) = f(x)$

Because the self-composition of a function $f(f(x)) = f^2(x)$ is also called “iteration”, the square root of a function is usually called its ***iterative root***.

$$\varphi^n(x) = f^m(x)$$

is solved by the ***fractional iterates*** of a function f :

$$\varphi(x) = f^{m/n}(x)$$

Generalized Iteration



The exponential notation of the iteration of functions $f^n(x)$ can be extended beyond integer exponents:

- f^1 means f
- f^n for positive integers n are the well known iterations of f
- f^0 denotes the identity function, $f^0(x) = x$
- f^{-1} is the inverse function of f
- f^{-n} is the n -th iteration of the inverse of f
- $f^{1/n}$ is the n -th iterative root of f
- $f^{m/n}$ is the m -th iteration of the n -th iterative root of f , the *fractional iterate* of f

The family $f^t(x)$ forms the *continuous iteration group* of f .

Within this the *translation equation* $f^{a+b}(x) = f^a(f^b(x))$ is satisfied.

Applied to the Fall Experiment



May the function $v_1 = f(v_0)$ describe the speed change after a fall from x_0 to x_1
Then for $x_0 = 0$ and $x_1 = 1$ (without loss of generality) the speed at any point x is

$$v_x = F(v_0, x) = f^x(v_0)$$

This represents the continuous embedding of a discrete (time or space) dynamical system.

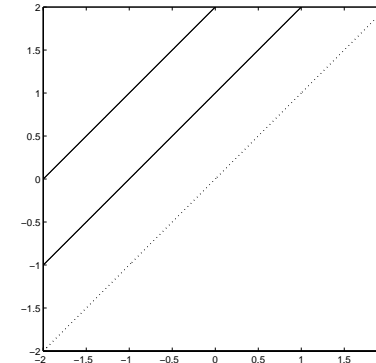
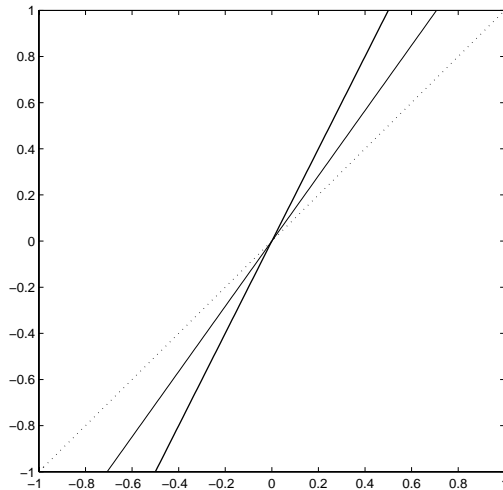
Mathematics so far provides no formal or numerical algorithm for finding embeddings in the general case.

Especially the speed in the middle $v_m = f^{1/2}(v_0)$ is given by the iterative root of f

Examples for Solutions of $\varphi^2 = f$

$$f(x) = x + 2 \Rightarrow \varphi(x) = x + 1$$

$$\text{Proof: } \varphi(\varphi(x)) = (x + 1) + 1 = x + 2 = f(x)$$

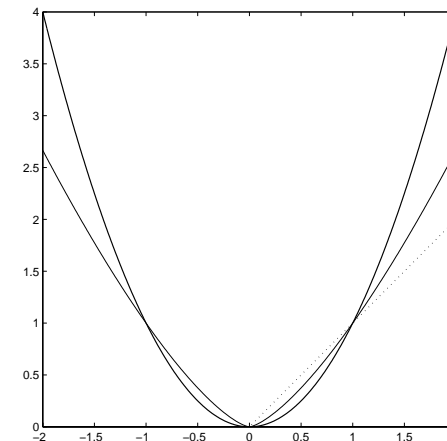


$$f(x) = 2x \Rightarrow \varphi(x) = \pm\sqrt{2}x$$

$$\text{Proof: } \varphi(\varphi(x)) = \sqrt{2}(\sqrt{2}x) = 2x = f(x)$$

$$f(x) = x^2 \Rightarrow \varphi(x) = |x|^{\sqrt{2}}$$

$$\text{Proof: } \varphi(\varphi(x)) = \left||x|^{\sqrt{2}}\right|^{\sqrt{2}} = x^2 = f(x)$$

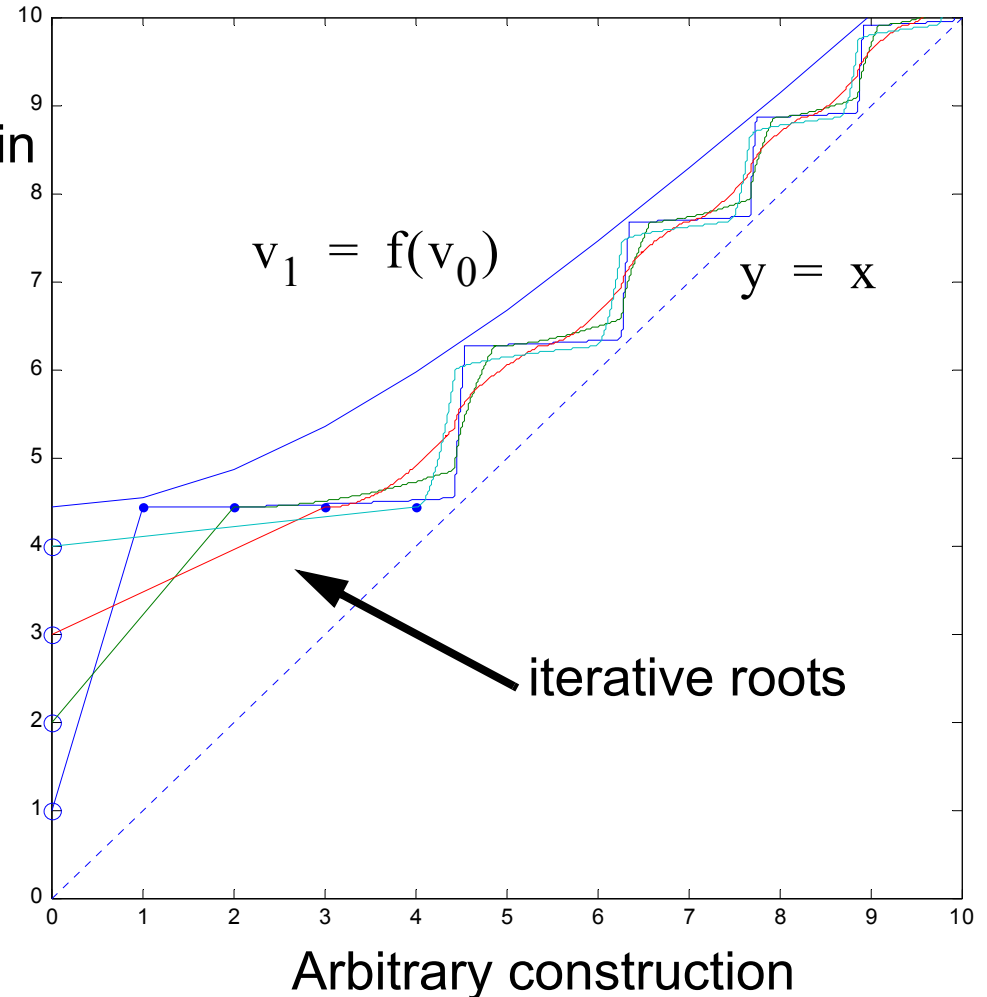


But try to figure out the iterative roots of $f(x) = x^2 + 1$ or $f(x) = e^x$!

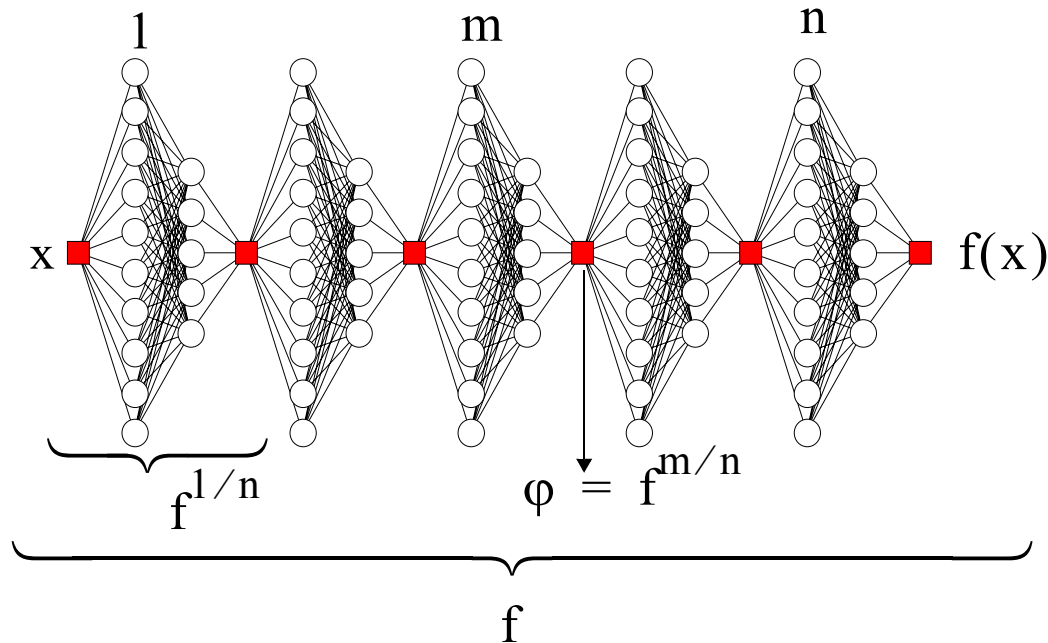
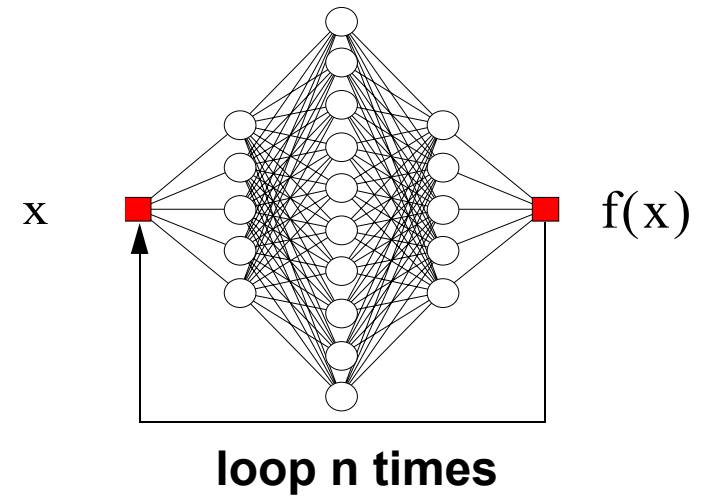
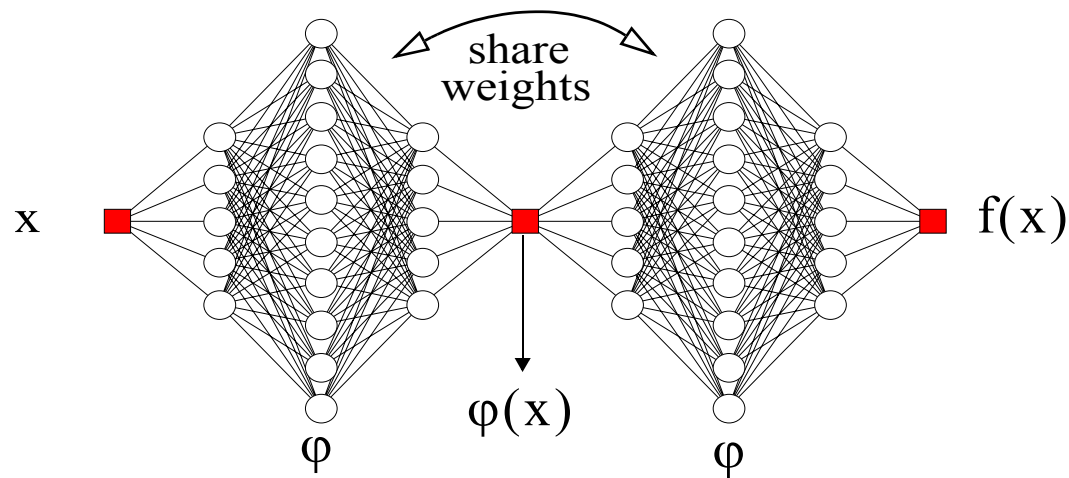
Properties of iterative roots

- Only very few functions have iterative roots
- Iterative roots are not unique in general
- Solutions strongly depend on the domain (continuous, monotonic, analytic)

=> Problem cannot be solved theoretically?



Mapping to Neural Networks



Training Algorithms



- **Weight Copy:** Train only the last layer and copy the weights continuously backwards
- **Weight Sharing:** Initialize corresponding weights with equal values and sum up all δw_i delivered by the network learning rule
- **Weight Coupling:** Start with different values and let the corresponding weights of the iteration layers approach each other by a term like $\delta w_i = \alpha(w_j - w_i)$
- **Regularization:** Add a penalty term to the error function which assigns an error to the weight-differences to regularize the network. This allows to utilize second order Training methods like quasi Newton for faster training.
- **Exact:** Compute the exact gradient of the iterated single-layer network

Layer Regularization

To enforce similar subnets add a penalty term to the errorfunction of the net:

$$E = E_{\text{app}} + E_{\text{reg}}$$

We use the sum of squared differences of corresponding weights in layers i, j
 m =number of weights per layer, n =number of layers:

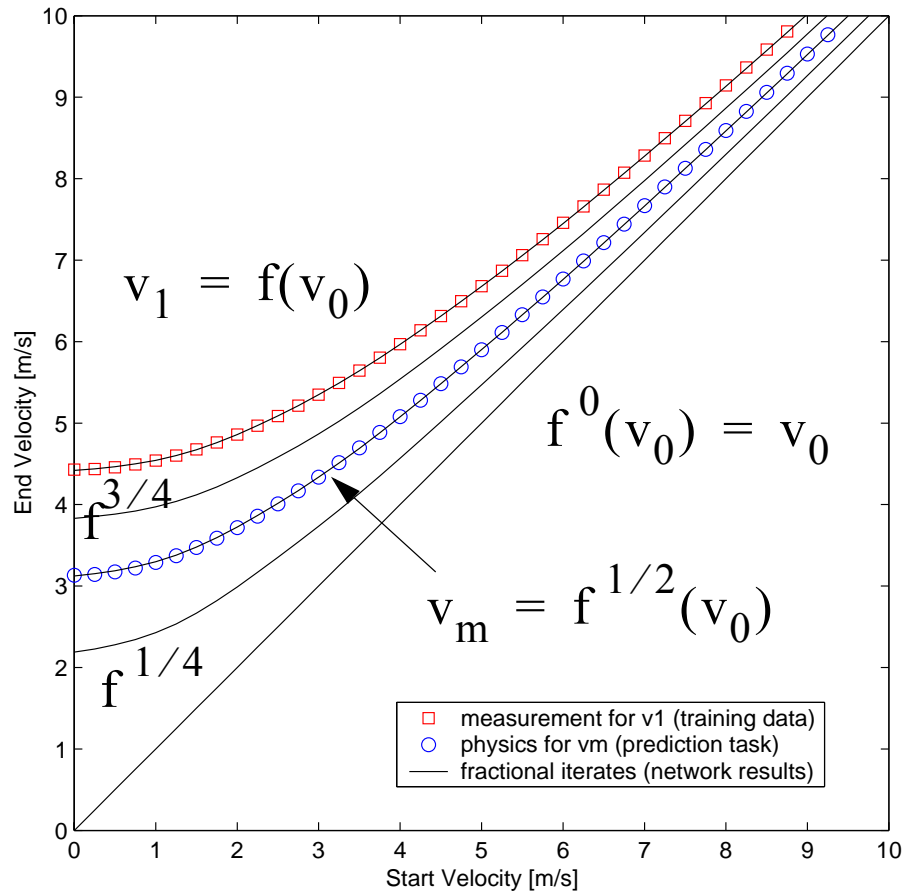
$$E_{\text{regsum}} = \sum_{i=1}^{n-1} \sum_{j=1}^{n-i} \sum_{k=i+1}^n (w_{i,j} - w_{i,k})^2 \text{ or normalized } E_{\text{regmean}} = \frac{E_{\text{regsum}}}{m \frac{n(n-1)}{2}}$$

To allow for some balance between regularization and approximation we introduce a parameter $0 < \alpha < 1$: $E = \alpha E_{\text{app}} + (1-\alpha) E_{\text{reg}}$

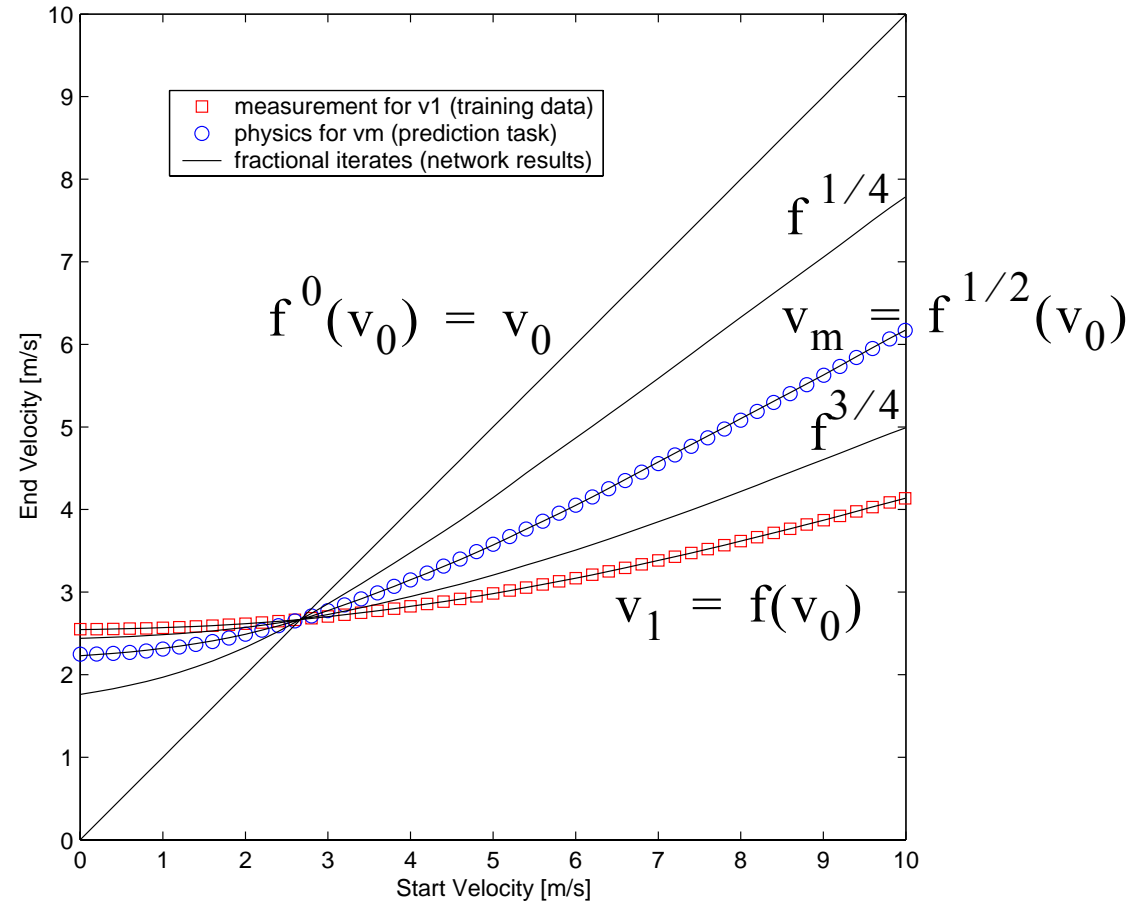
The derivation term of the penalty term looks like $\frac{\partial E_{\text{regsum}}}{\partial w_{i,j}} = 2 \sum_{k=1}^n (w_{i,j} - w_{i,k})$

Network Results for the “Fall”

no friction



with friction

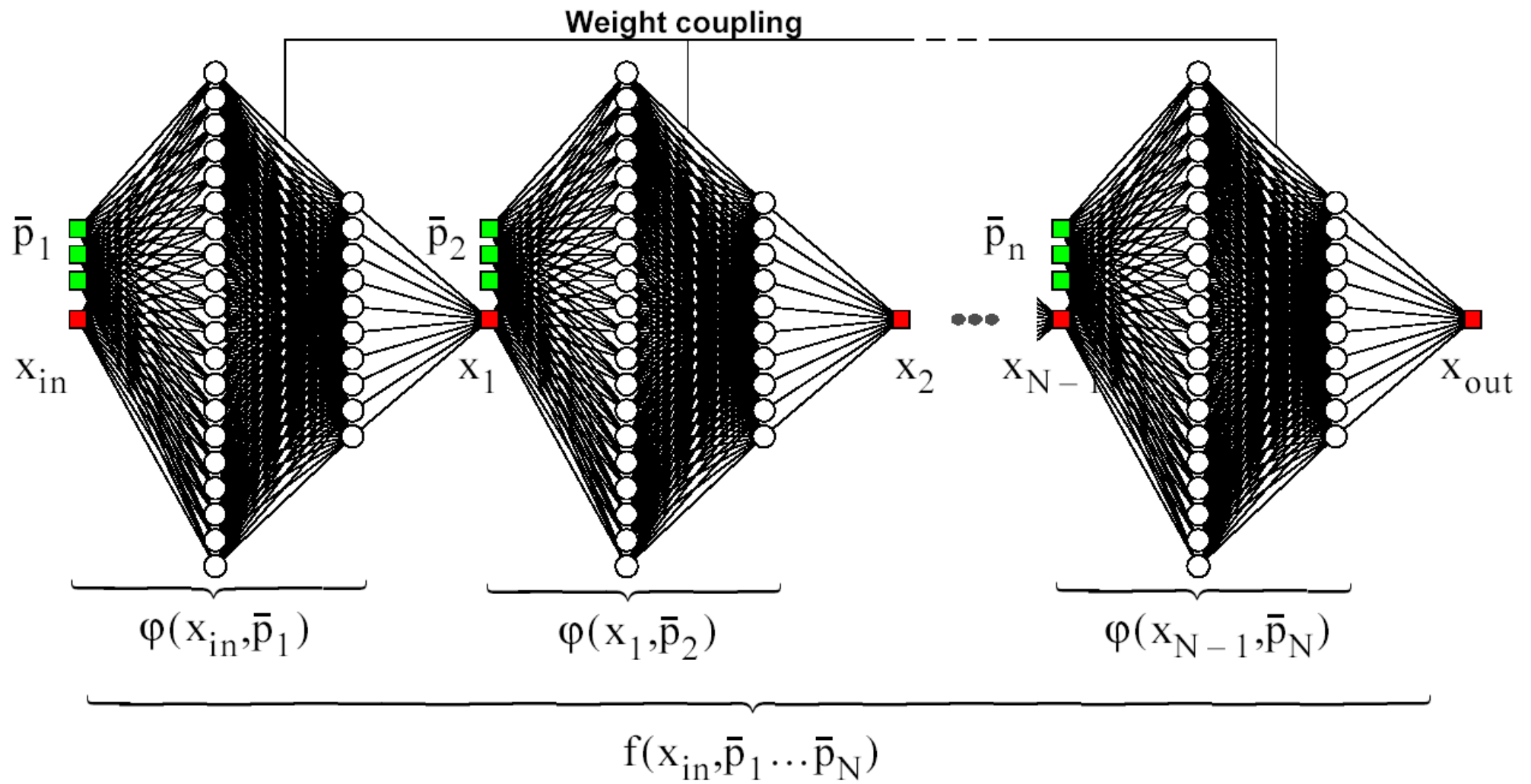


The Network results are conform to the laws of physics up to a mean error of 10^{-6}

Industrial Application:



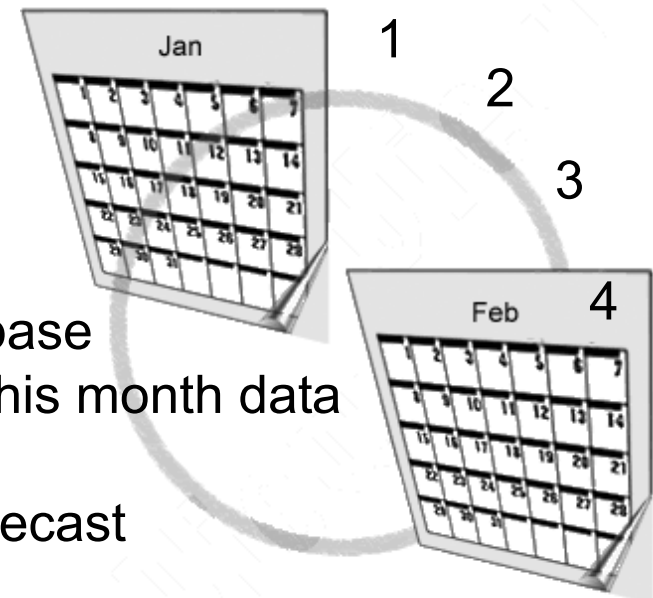
Modelling a Steel Mill



Weekly forecast from monthly data

Situation:

- Lot of data has been collected for years on a monthly base
- There is a good model for predicting next month from this month data
- Speed of market has increased, demand for weekly forecast



Solution:

The weekly model is the 4-th root of the monthly model!

Conclusion



Fractional Iterates computed by neural networks can compute otherwise hardly accessible problems

Outlook

Similar methods can be applied to different functional equations representing other abstract principles as well

Visit me!

Find a MATLAB Toolbox and lots of literature at www.reglos.de/kindermann/ffx.html