Deep learning to diagnose the neutron star

Kenji Fukushima

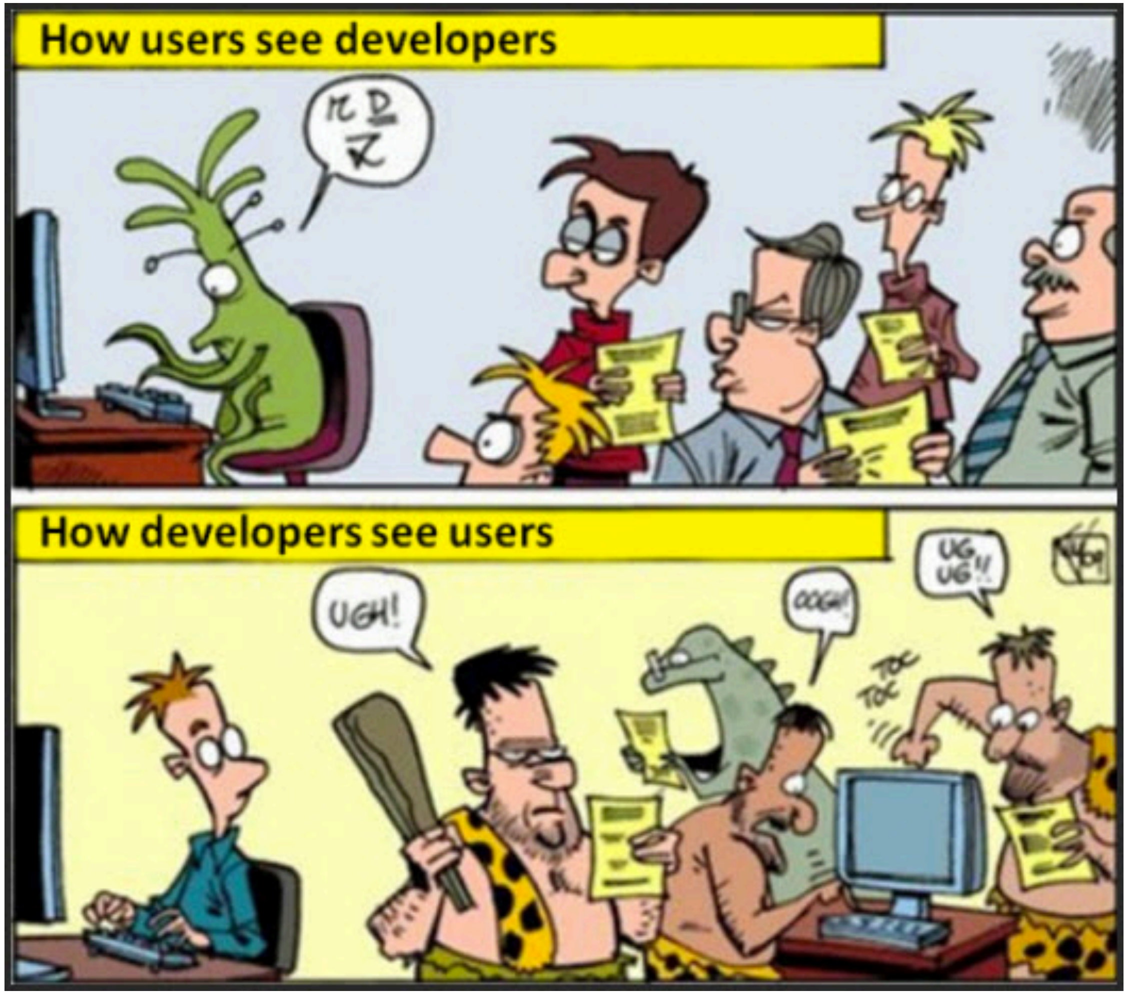
The University of Tokyo

Based on work with Yuki Fujimoto, Koichi Murase

— Deep Learning and Physics 2018 —

Disclaimer

I am a “user” of the deep learning…



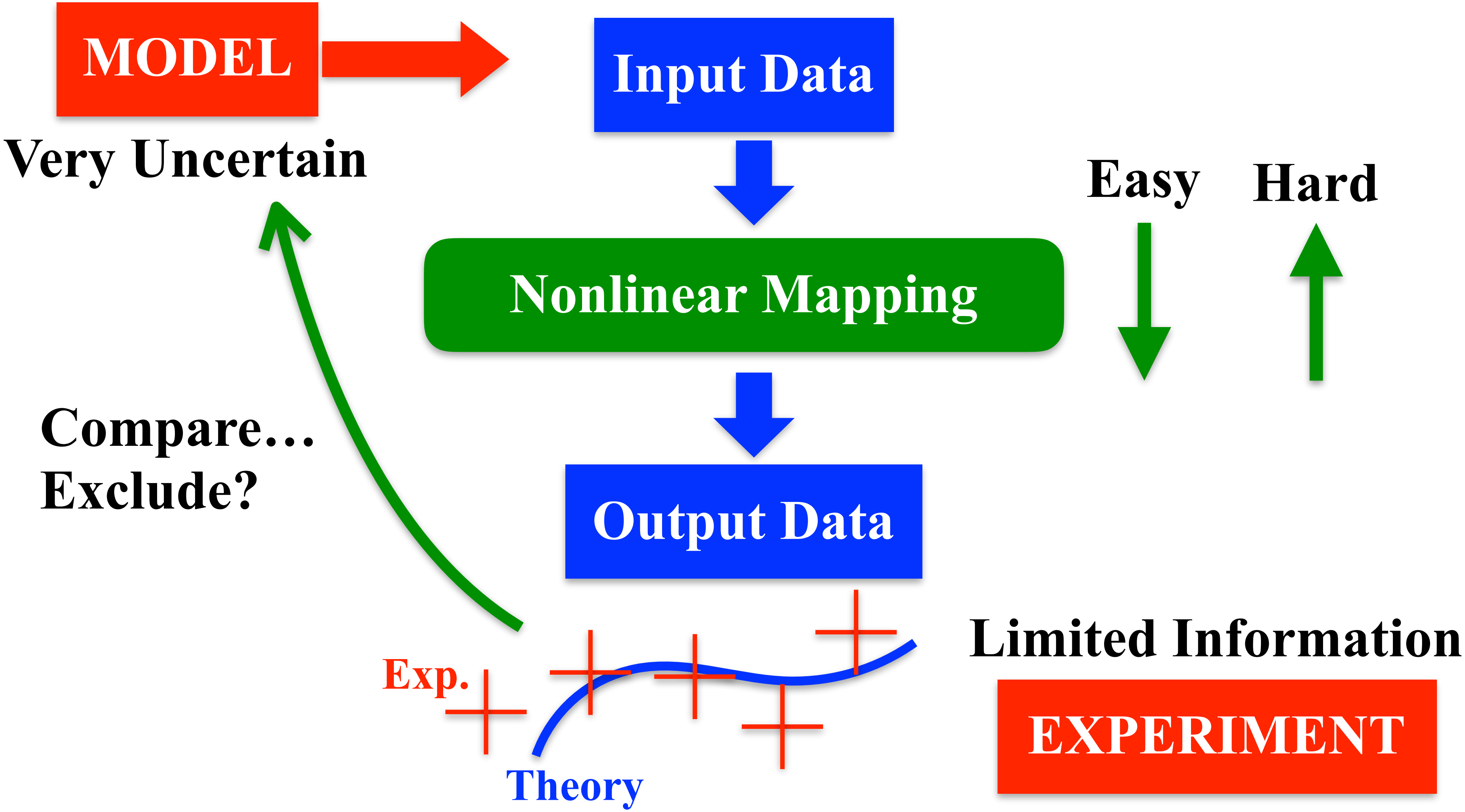
From the point of view of physics users…

Sounds fancy is not enough…

Useful ?

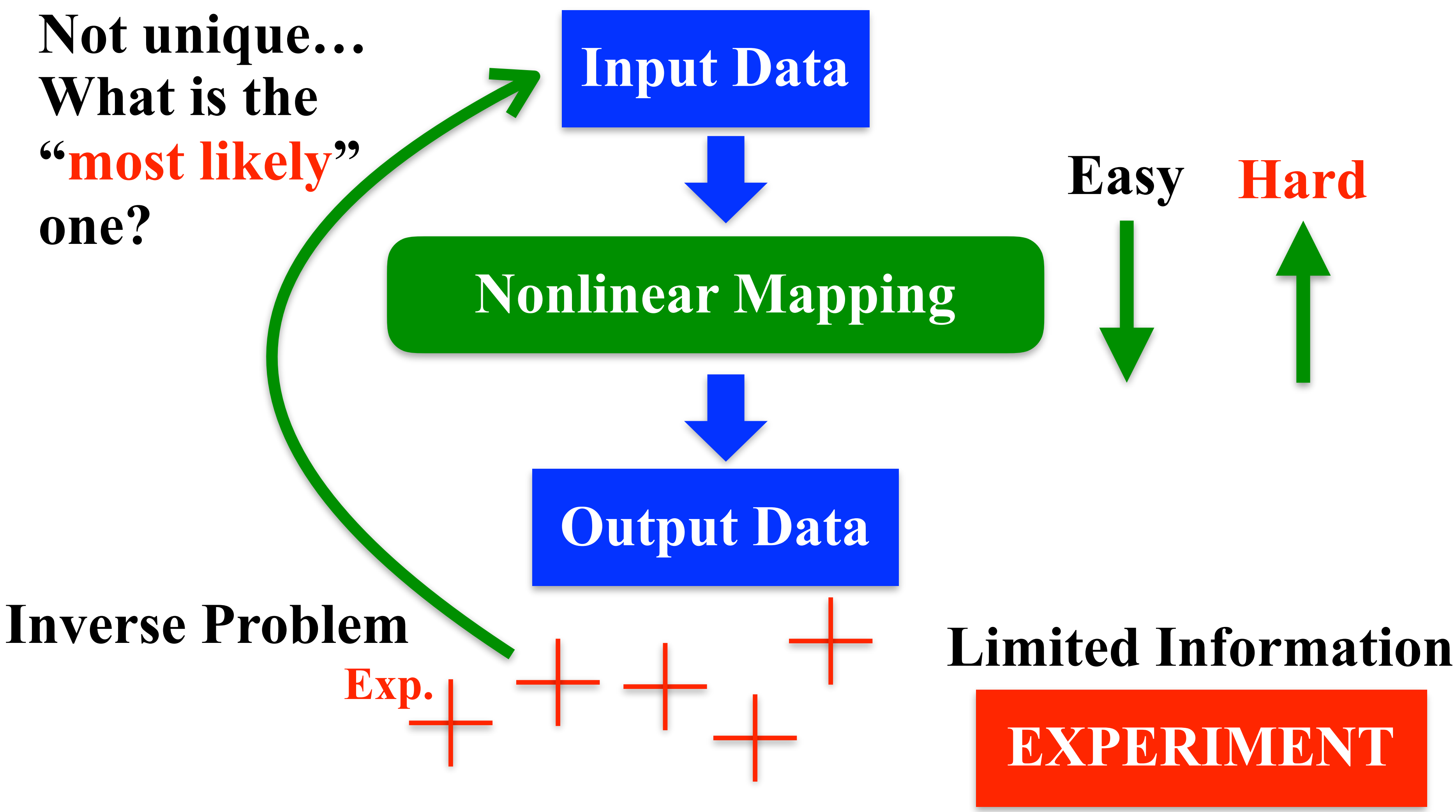
Advantageous?

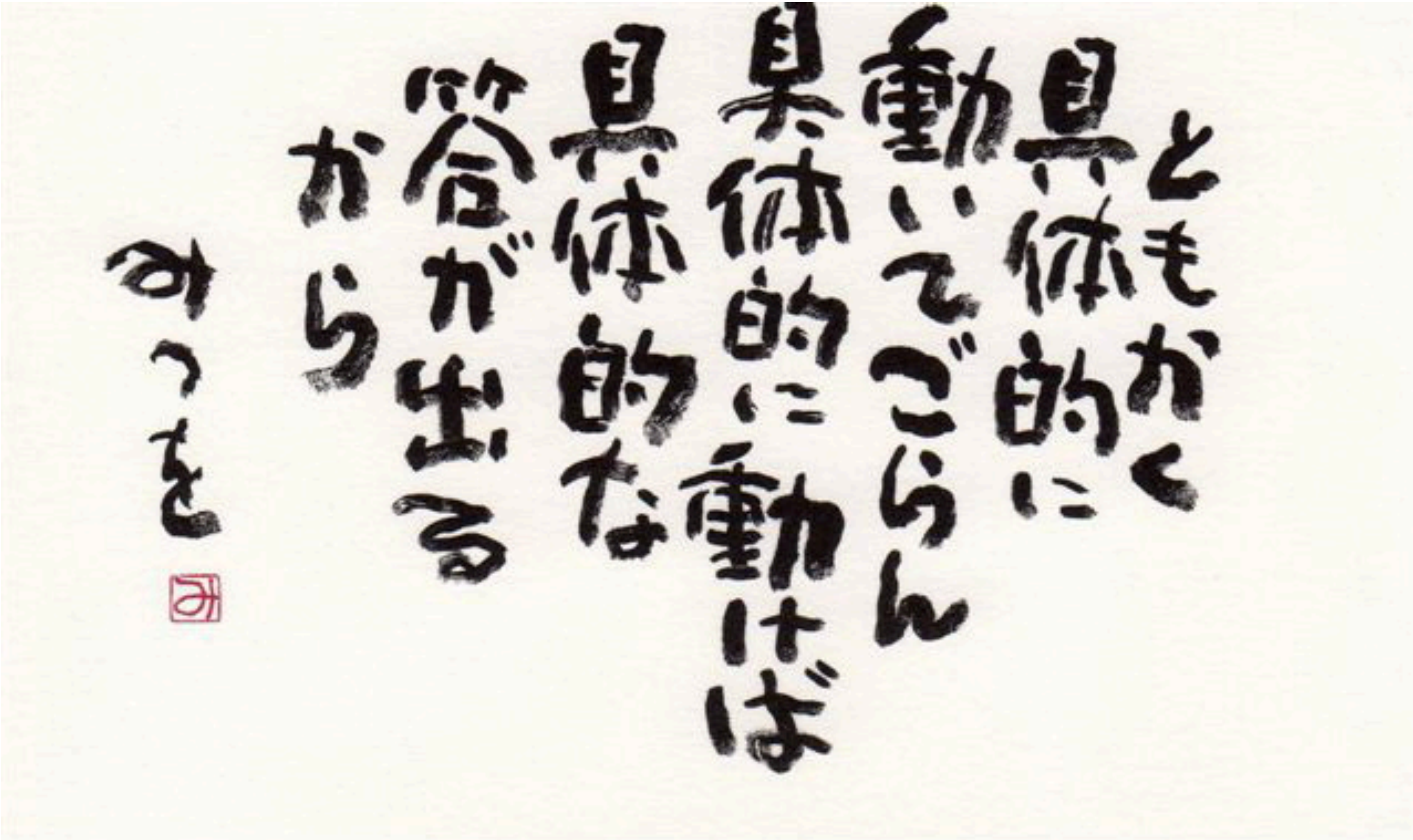
Conventional Physics Approach



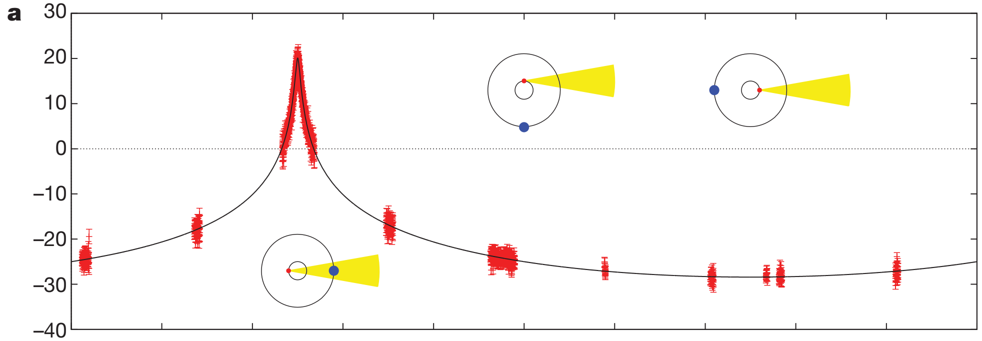
One another to infinity, so what ?

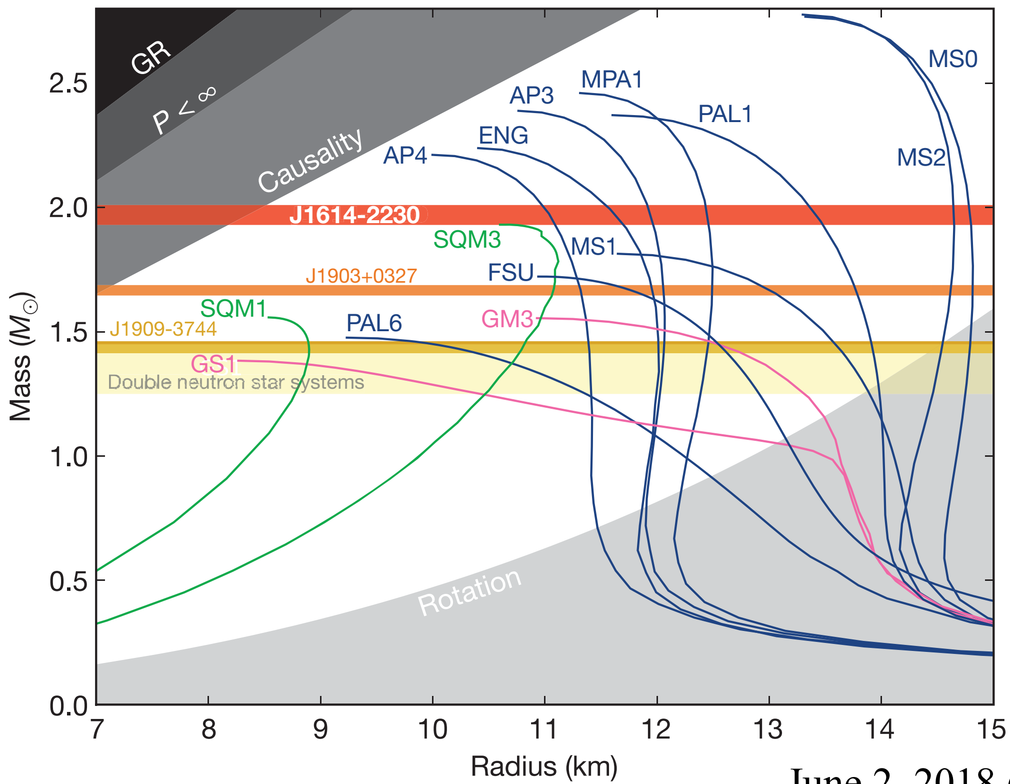
"Model Independent" Analysis




ともかく
具体的に
動いてごらん
具体的に動けば
具体的な
答が出る
から
みつを
み

Neutron Star EoS





Demorest et al. (2010)

Precise determination of NS mass using Shapiro delay

1.928(17) M_{sun} (J1614-2230)

(slightly changed in 2016)

Antoniadis et al. (2013)

2.01(4) M_{sun} (PSRJ0348+0432)

Neutron Star EoS

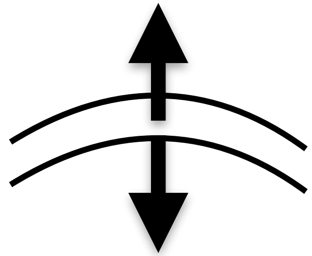
Equation of State **Input Data**

Nonlinear Mapping

Pressure : p
Mass density : ρ
$(\text{Energy density}:\ \varepsilon = \rho c^2)$

$$p = p(\rho)$$

Tolman-Oppenheimer--Volkoff (TOV) Eqs



M-*R* Relation **Output Data**

NS mass : M
NS radius : R

$$M = M(\rho_{\max})$$
$$R = R(\rho_{\max})$$

Mathematically one-to-one correspondence

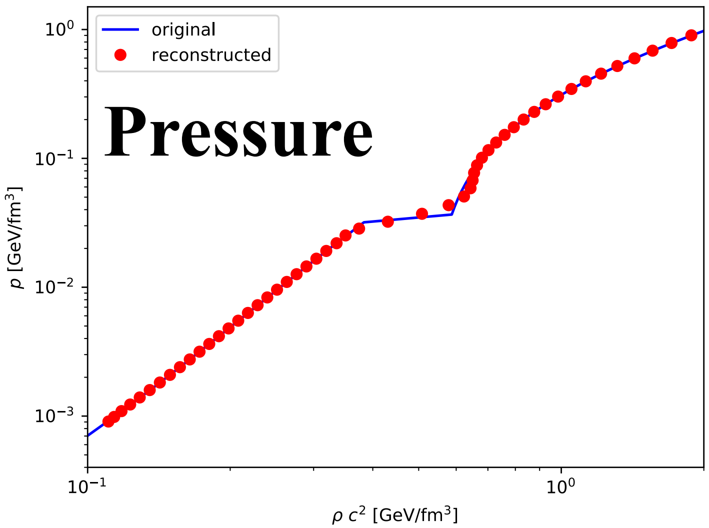
Neutron Star EoS

Lindblom (1992)

Brute-force solution of the inverse problem

Test data put by hand

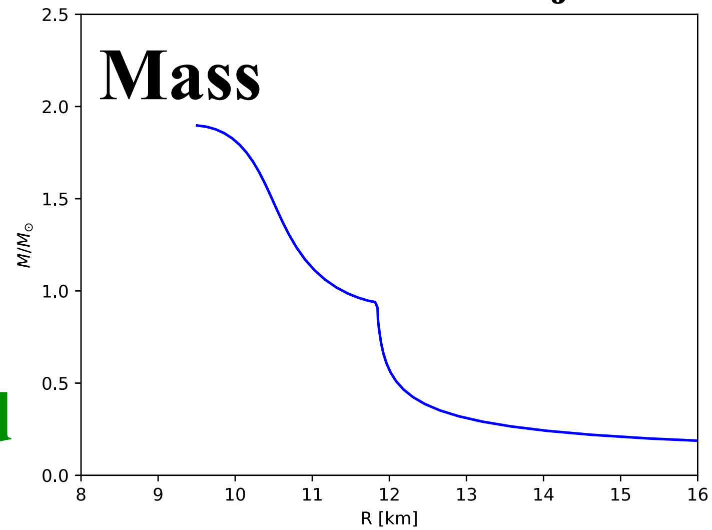
Thanks to Y. Fujimoto



Density

Solve TOV

Reconstructed



Radius

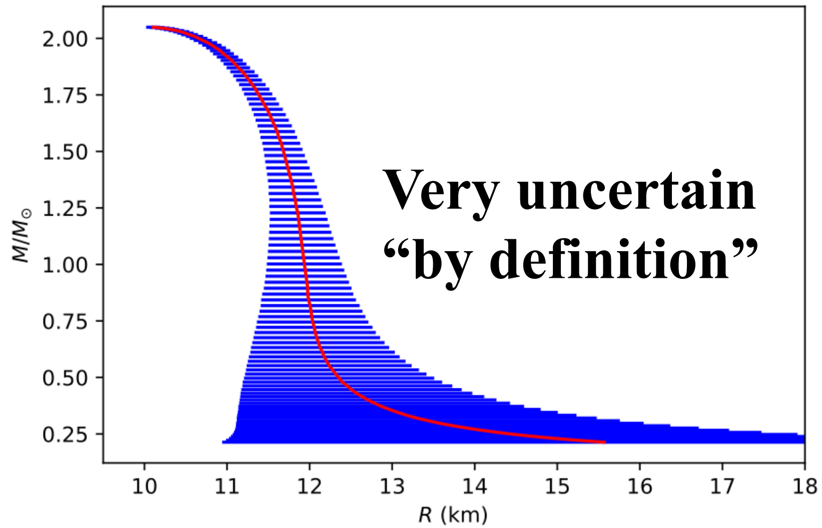
The answer exists!

No magic box…
Only “solvable” problem can be solved…

Neutron Star EoS (Side Remark)

***R* is fixed by TOV with *p*(*R*)=0   ("surface" condition)**

$$dp/dr(r = R) = 0$$
$$d^2p/dr^2(r = R) \propto M^2/R^2$$



Determination of *R* is very uncertain, and on top of that, *R* itself is anyway very uncertain…

People do not care assuming that NS mass > 1.2 M_{sun}

Model Independent Analysis

Bayesian Analysis

B : M-R Observation
A : EoS Parameters

(Bayes' theorem) Normalization

$$\underbrace{P(A|B)}_{\text{Want to know}}\cancel{P(B)} = \underbrace{P(B|A)}_{\text{Likelihood}}\underbrace{P(A)}_{\text{prior}}$$

Calculable by TOV — Model

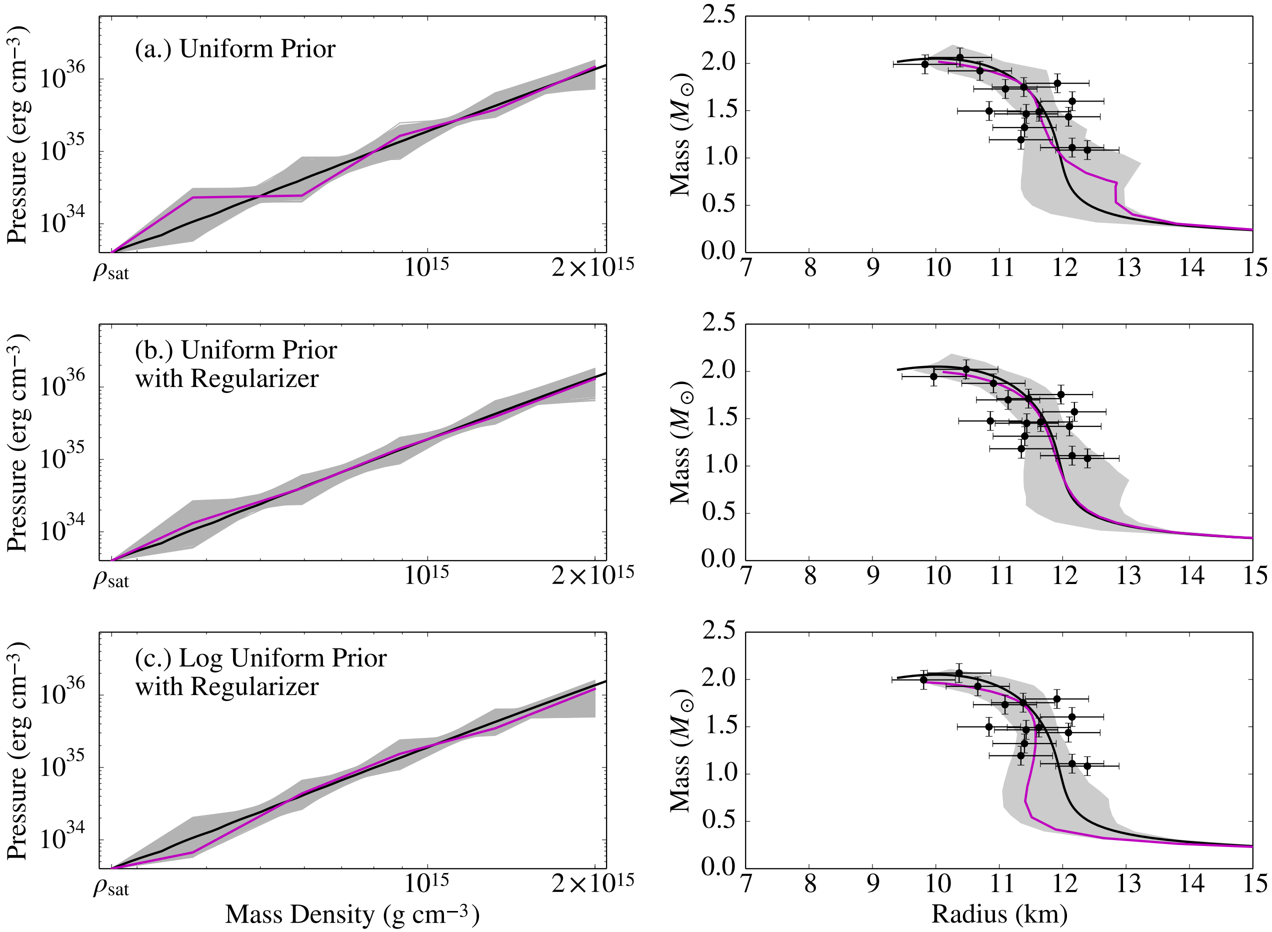
Model must be assumed.
EoS parametrization must be introduced.
Integration in parameter space must be defined.

If infinite observations, prior dependence should be gone.

Model Independent Analysis

Raithel-Ozel-Psaltis (2017) Mock data (SLy + Noises)

Prior Dep.



Black curve
True EoS

Magenta curve
Guessed EoS

Gray band
68% credibility

Model Independent Analysis

Bayesian Analysis
Supervised Learning

Several *M-R* observation points with errors

Nonlinear Mapping

Several parameters to characterize EoS

$$\{M_i, R_i\} \qquad \{P_i\} = F(\{M_i, R_i\}) \qquad \{P_i\}$$

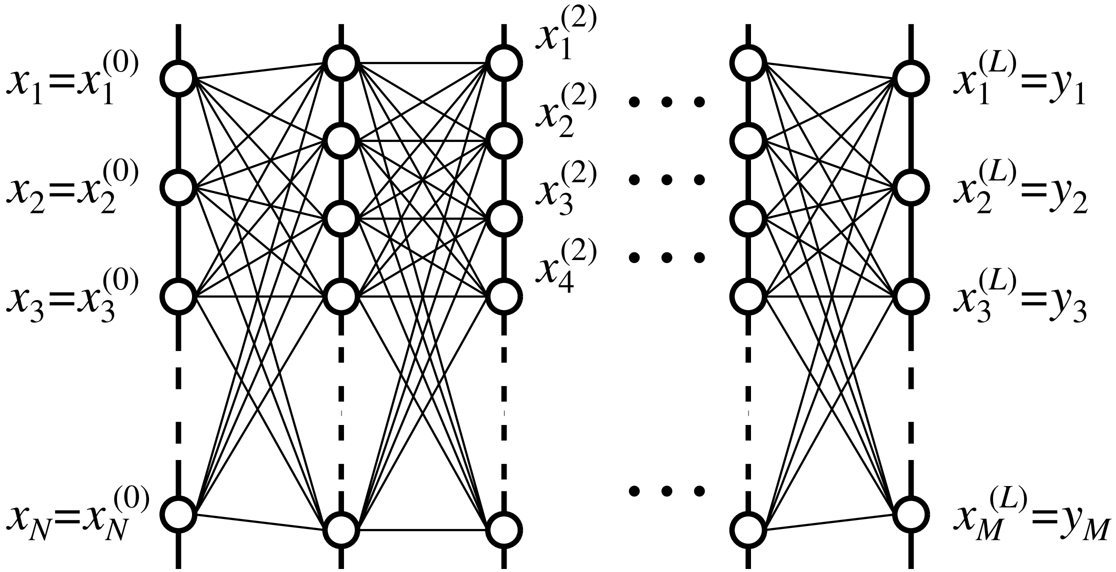
~ **15 Points (observations)** ~ **5 Points**

Too precise parametrization of EoS is useless (beyond the uncertainty from observations)

Deep Learning

$\{M_i, R_i\}$

~ 15



$\{P_i\}$

~ 5

$$x_i^{(k+1)} = \sigma^{(k+1)}\left(\sum_{i=1}^{N_k} \underline{W_{ij}^{(k+1)}}\, x_j^{(k)} + \underline{a_i^{(k+1)}}\right)$$

Backpropagation

Parameters to be tuned

~~sigmoid func.~~	ReLU	tanh
$\cancel{\sigma(x) = 1/(e^x + 1)}$	$\sigma(x) = \max\{0, x\}$	$\sigma(x) = \tanh(x)$

Deep Learning

Our Neural Network Design

Layer index	Nodes	Activation
1	30	N/A
2	60	ReLU
3	40	ReLU
4	40	ReLU
5	5	tanh

Probably we don’t need such many hidden layers and such many nodes… anyway, this is one working example…

Deep Learning

For good learning, the "textbook" choice is important…

Training data (200000 sets in total)

Randomly generate **5** sound velocities → EoS × 2000 sets

Solve TOV to identify the corresponding *M*-*R* curve

Randomly pick up **15** observation points × (n_s = 100) sets

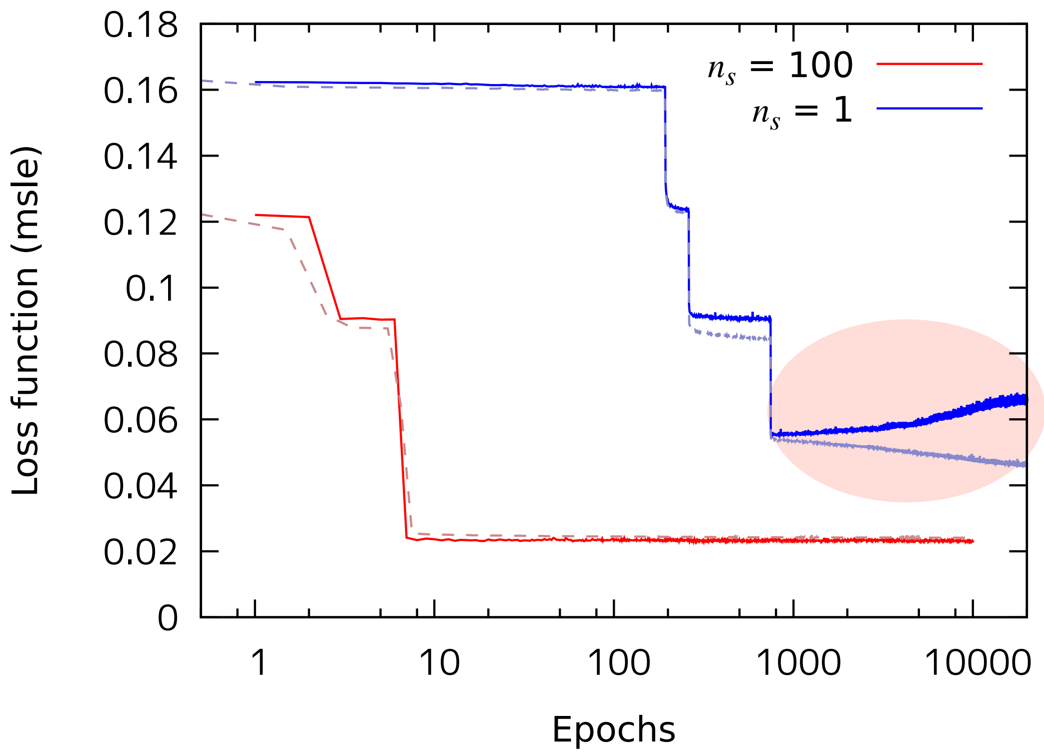
(with $\Delta M = 0.1 M_\odot$, $\Delta R = 0.5\,\mathrm{km}$)

The machine learns the *M*-*R* data have error fluctuations

Validation data (200 sets)

Generate independently of the training data

Deep Learning



"Loss Function"
= deviation from the
true answers (**msle**)

Monotonically decrease for the training data, but not necessarily so for the validation data

With fluctuations in the training data, the learning goes quickly

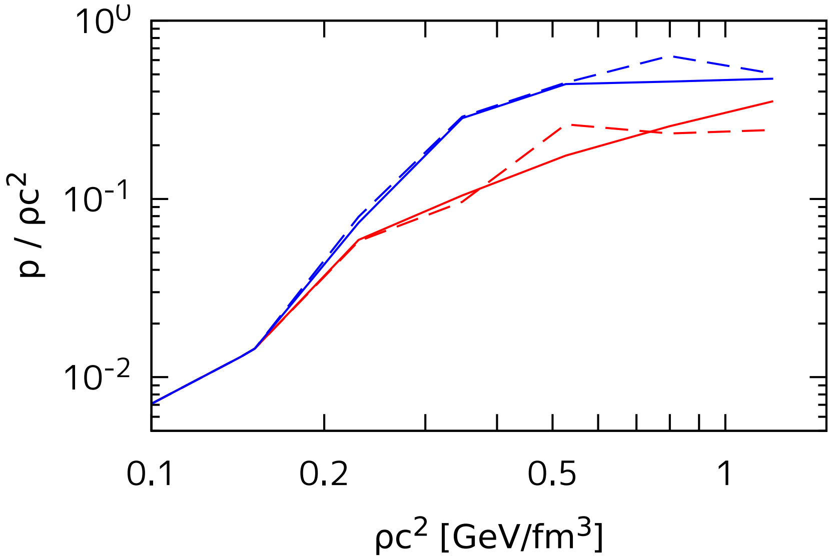
Once the over-fitting occurs,
the model becomes more stupid…

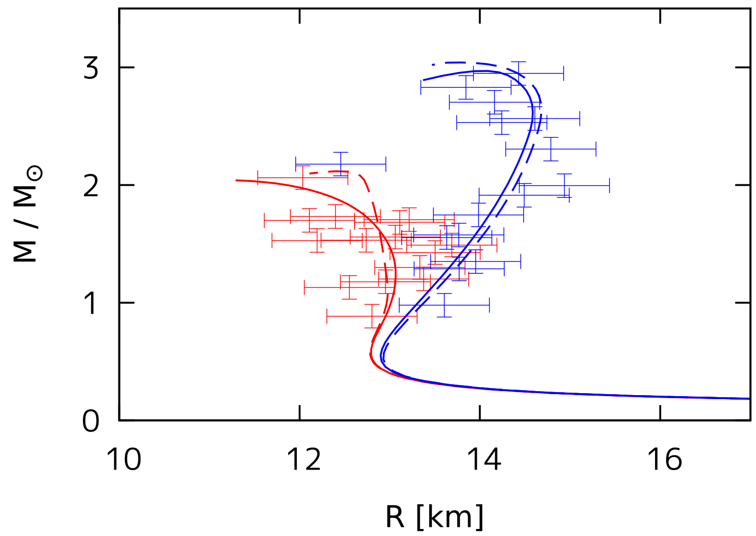
Deep Learning

Test with the validation data **Fujimoto-Fukushima-Murase (2017)**

(parameters not optimized to fit the validation data)

Two Typical Examples (not biased choice)





: randomly generated original EoS

: reconstructed EoS and associated *M*-*R*

Deep Learning

Overall performance test

Mass ($M_\odot$)	0.6	0.8	1.0	1.2	1.4	1.6	1.8
RMS (km)	0.16	0.12	0.10	0.099	0.11	0.11	0.12

(with $\Delta M = 0.1 M_\odot$, $\Delta R = 0.5\,\mathrm{km}$)

Very promising!

Credibility estimate has not been done for simplicity, but it can be included in the learning process.

Usefulness confirmed, easy implementation but advantageous ?

Bayesian or NN, which to choose?

Baysian vs NN

EoS $\boldsymbol{\theta} := \{c_{s,i}^2\}$ **Obs.** $\mathcal{D} = \{(M_i, R_i)\}$

Bayesian $f_{\mathrm{MAP}}(\mathcal{D}) = \arg\max_{\boldsymbol{\theta}} [\underbrace{\Pr(\boldsymbol{\theta}) \Pr(\mathcal{D}|\boldsymbol{\theta})}_{\Pr(\boldsymbol{\theta}|\mathcal{D})}]$

NN minimizes $\langle \ell[f] \rangle = \int \underline{d\boldsymbol{\theta}} d\mathcal{D} \, \underline{\Pr(\boldsymbol{\theta}) \Pr(\mathcal{D}|\boldsymbol{\theta})} \ell(\boldsymbol{\theta}, f(\mathcal{D}))$

Approximated estimated → Baysian

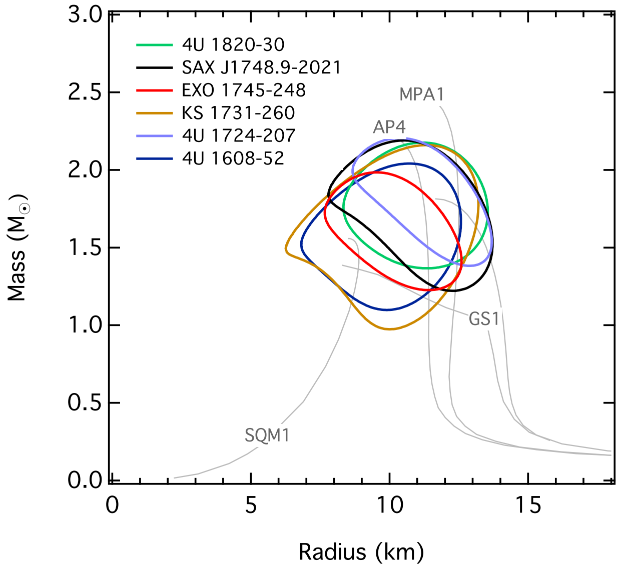
NN allows for more general choice of loss functions.
Baysian assumes parametrized likelihood functions.

Conclusion

Yes, useful!

Maybe, less biased ?

Developing a toolkit for real data like



not discrete data with error,
but regions of credibility

Error analysis (credibility estimate)
in the output side