The R-mAtrIx Net

Evgeny Sobko

ÉCOLE NORMALE SUPÉRIEURE

based on

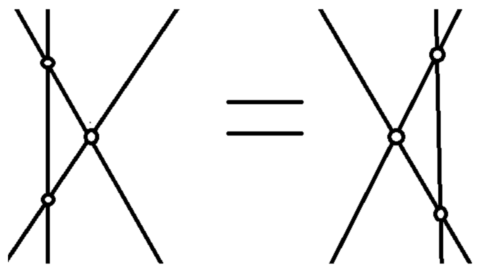
Shailesh Lal, Suvajit Majumder, ES arxiv: 2304.07247 & work in progress

IGST 2023, Zürich

Plan

- Neural Networks. Overview
- Integrable Spin Chains. Overview
- Machine Learning the R-Matrix
- Exploring the space of Integrable spin chains
- Future directions

- Bootstrap approach : make focus on symmetries & self consistency
- (S-matrix , Conformal, Matrix) Bootstrap
- S-matrix Bootstrap = analyticity + unitarity +crossing+…
- Integrable Bootstrap = S-matrix Bootstrap + Factorisation $\rightarrow$ Exact solution



- Practically one have to solve a system of functional equations in the certain functional space. In general it is a non-convex problem.

Neural Networks

- Usually we think about functions as elements of Hilbert space and make approximation by summation (fourier, orthogonal polynomials, etc):

$$f = f_1 + f_2 + f_3 + ...$$

- Another idea : use composition instead of sum

$$f = f_1 \circ f_2 \circ f_3 \circ ...$$

- Namely let's consider the following map $a^{out}(a^{in}) : \ \mathbb{R}^{n_0} \to \mathbb{R}^{n_{L+1}}$

$$a^{out} = A^{L+1} \circ h \circ A^L \circ ... \circ h \circ A^1 \circ a^{in}$$

alternating affine transforms $A^{(l)}$ and a certain nonlinear map(activation function) *h*

$$\begin{pmatrix} a_1^{(\ell)} \\ a_2^{(\ell)} \\ \vdots \\ a_{n_\ell}^{(\ell)} \end{pmatrix} = h \left[\begin{pmatrix} w_{1,0}^{(\ell)} & w_{1,1}^{(\ell)} & \dots & w_{1,n_{\ell-1}}^{(\ell)} \\ w_{2,0}^{(\ell)} & w_{2,1}^{(\ell)} & \dots & w_{2,n_{\ell-1}}^{(\ell)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n_\ell,0}^{(\ell)} & w_{n_\ell,1}^{(\ell)} & \dots & w_{n_\ell,n_{\ell-1}}^{(\ell)} \end{pmatrix} \begin{pmatrix} a_1^{(\ell-1)} \\ a_2^{(\ell-1)} \\ \vdots \\ a_{n_{\ell-1}}^{(\ell-1)} \end{pmatrix} + \begin{pmatrix} b_1^{(\ell)} \\ b_2^{(\ell)} \\ \vdots \\ b_{n_\ell}^{(\ell)} \end{pmatrix} \right], \qquad h \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix} = \begin{pmatrix} h(z_1) \\ h(z_2) \\ \vdots \\ h(z_n) \end{pmatrix}$$

$$a^{(0)} = a^{in}, \quad a^{(L+1)} = a^{out}$$

Popular activation functions h(x):

$$\text{relu}(x) = \max(0, x), \ \text{swish}(x) = \frac{x}{1 + e^{-x}}, \ \tanh(x), ...$$

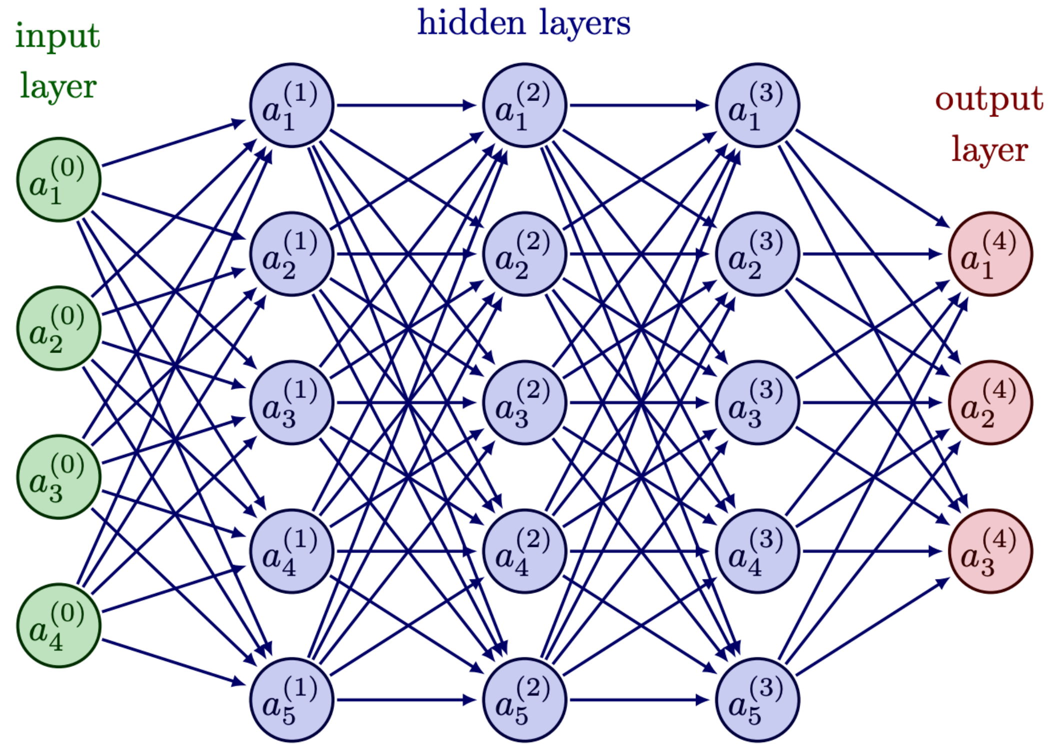

input layer
hidden layers
output layer
$a_1^{(0)}$ $a_2^{(0)}$ $a_3^{(0)}$ $a_4^{(0)}$
$a_1^{(1)}$ $a_2^{(1)}$ $a_3^{(1)}$ $a_4^{(1)}$ $a_5^{(1)}$
$a_1^{(2)}$ $a_2^{(2)}$ $a_3^{(2)}$ $a_4^{(2)}$ $a_5^{(2)}$
$a_1^{(3)}$ $a_2^{(3)}$ $a_3^{(3)}$ $a_4^{(3)}$ $a_5^{(3)}$
$a_1^{(4)}$ $a_2^{(4)}$ $a_3^{(4)}$

Figure 1: The schematic for a Dense Neural Network, also known as a Fully Connected neural network.

Classical results *Cybenko, 1989; Hornik et al., 1989,* and their variations demonstrate that a neural network with fixed depth (one hidden layer is enough!) and width going to infinity can uniformly approximate any continuous function on a compact set, if the activation function is continuous and nonpolynomial.

Many versions of the universal approximation theorem!

Various functional spaces, classes of activation functions, depth vs width etc etc

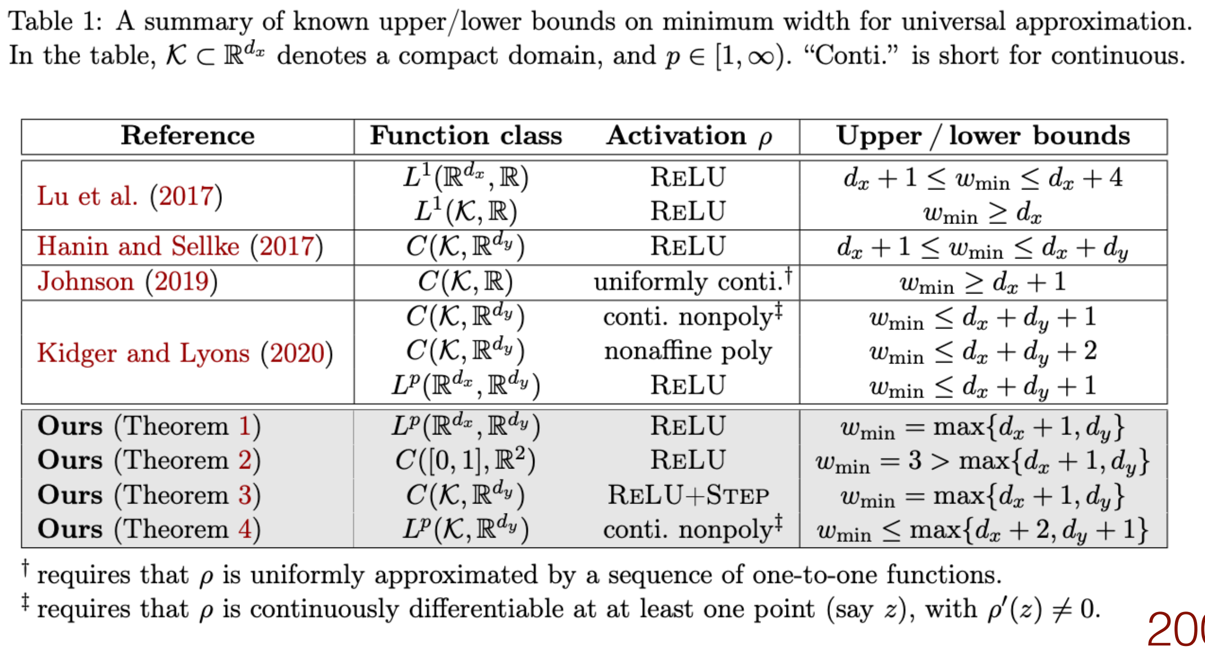


Table 1: A summary of known upper/lower bounds on minimum width for universal approximation. In the table, $\mathcal{K} \subset \mathbb{R}^{d_x}$ denotes a compact domain, and $p \in [1, \infty)$. "Conti." is short for continuous.

Reference	Function class	Activation ρ	Upper / lower bounds
Lu et al. (2017)	$L^1(\mathbb{R}^{d_x}, \mathbb{R})$	ReLU	$d_x + 1 \leq w_{\min} \leq d_x + 4$
	$L^1(\mathcal{K}, \mathbb{R})$	ReLU	$w_{\min} \geq d_x$
Hanin and Sellke (2017)	$C(\mathcal{K}, \mathbb{R}^{d_y})$	ReLU	$d_x + 1 \leq w_{\min} \leq d_x + d_y$
Johnson (2019)	$C(\mathcal{K}, \mathbb{R})$	uniformly conti.†	$w_{\min} \geq d_x + 1$
Kidger and Lyons (2020)	$C(\mathcal{K}, \mathbb{R}^{d_y})$	conti. nonpoly‡	$w_{\min} \leq d_x + d_y + 1$
	$C(\mathcal{K}, \mathbb{R}^{d_y})$	nonaffine poly	$w_{\min} \leq d_x + d_y + 2$
	$L^p(\mathbb{R}^{d_x}, \mathbb{R}^{d_y})$	ReLU	$w_{\min} \leq d_x + d_y + 1$
Ours (Theorem 1)	$L^p(\mathbb{R}^{d_x}, \mathbb{R}^{d_y})$	ReLU	$w_{\min} = \max\{d_x + 1, d_y\}$
Ours (Theorem 2)	$C([0,1], \mathbb{R}^2)$	ReLU	$w_{\min} = 3 > \max\{d_x + 1, d_y\}$
Ours (Theorem 3)	$C(\mathcal{K}, \mathbb{R}^{d_y})$	ReLU+Step	$w_{\min} = \max\{d_x + 1, d_y\}$
Ours (Theorem 4)	$L^p(\mathcal{K}, \mathbb{R}^{d_y})$	conti. nonpoly‡	$w_{\min} \leq \max\{d_x + 2, d_y + 1\}$

† requires that ρ is uniformly approximated by a sequence of one-to-one functions.
‡ requires that ρ is continuously differentiable at at least one point (say z), with $\rho'(z) \neq 0$.

2006.08859

Learning

- Neural network learns a target function $f(x)$ of the input data x by optimizing a cost function $\mathcal{L}$ which provides a measure of the discrepancy between the actual and desired properties of the function $f(x)$.

- The simplest example is a *supervised learning* = approximation to the given data $\mathcal{D} = \{(x, y)\}$:

$$f\left(x\right) \approx y \qquad \forall \qquad (x, y) \in \mathcal{D}$$

which can be done by minimising $\mathcal{L}$ w.r.t. weights and biases $(\mathbf{w}, \mathbf{b})$:

$$\mathcal{L}\left(\{\mathbf{w}, \mathbf{b}\}\right) = \sum_{(x,y)\in\mathcal{D}} \left|y - f\left(\mathbf{w}, \mathbf{b}; x\right)\right|^q \xrightarrow[\mathbf{w},\mathbf{b}]{} min$$

inputs and outputs can be whatever you want : values of Bessel functions, photos of cats and dogs, Calabi-Yau and their Hodge numbers etc

- In *unsupervised learning*, neural networks strive to satisfy a specific condition by minimizing the loss function encoding that condition.

  For example one can find approximate solution of ODE $y'(x) = y(x)\cos x,\ y(0) = 1$ on the interval (-1,1) minimising the loss :

$$\mathcal{L}\left(\{\mathbf{w},\mathbf{b}\}\right) = \sum_{x\in(-1,1)} |f'(x) - f(x)\cos x|^2 + |f(0)-1|^2 \xrightarrow[\mathbf{w},\mathbf{b}]{} min$$

- The fantastic success of NNs in real life! Maybe you've grown weary from the abundance of news surrounding its achievements or maybe you feel AI anxiety and can't stop to read

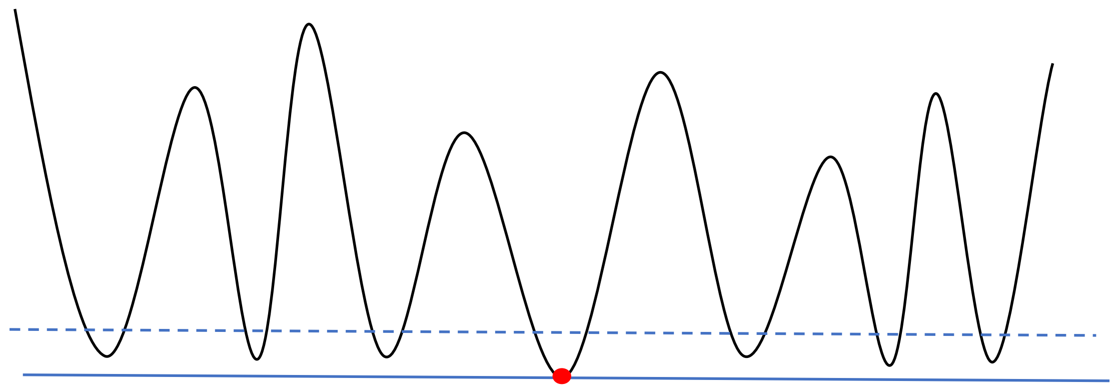
- Math and theoretical physics : Watch M.Douglas "How will we do mathematics in 2030?"

  - String Landscape and related Algebraic Geometry
    Y.-H. He '17, Krefl, Seong '17, Ruehle '17, Carifio, Halverson, Krioukov, Nelson '17 + hundreds others
  - nonlinear PDE, ODE, etc a lot of papers
  - some results in Conformal Bootstrap and Matrix Models
  - very few papers on classical integrability
    Liu, Tegmark'21'22, Bondesan, Lamacraft '19, Krippendorf, Lust, Syvaeri '21
  - and nothing so far about quantum integrability!

Why do <u>deep</u> neural networks work?!

there is no good understanding but there is some intuition from toy models and experiments

Choromanska, Henaff, Mathieu, Arous, LeCun '15 analysed very simplifyied version of NN and using similarity to spherical spin glasses (*Auffinger et all '10*) have shown the following structure of minima :



Auffinger, A., Ben Arous, G., and Cerny, J. '10 used Random Matrices to analyse the distribution of critical points in spin glasses :

$$H_{N,p}(\boldsymbol{\sigma}) = \frac{1}{N^{(p-1)/2}} \sum_{i_1,\ldots,i_p=1}^{N} J_{i_1,\ldots,i_p} \sigma_{i_1} \ldots \sigma_{i_p},$$

$$\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_N) \in S^{N-1}(\sqrt{N})$$

The similar picture people see from numerical analysis.
Dauphin, Pascanu, Gulcehre, Cho, Ganguli, Bengio '14 etc

Extensive work on new physically inspired algorithms for optimazation see e.g. last works on Energy-Conserving Descent (*De Luca, E.Silverstein), Gatti ('22)'23*

Spin-chains. Ultra quick overview

- Hilbert space : $\mathbb{V} = V_1 \otimes ... \otimes V_L, \quad V_i \sim V = \mathbb{C}^d$
- Hamiltonian : $H = \sum_{i=1}^{L} H_{i,i+1}, \qquad H_{L,L+1} \equiv H_{L,1}$
- Example: spin-1/2 XYZ magnet : $H = \sum_{i=1}^{L} \sum_{\alpha} J^\alpha S_i^\alpha S_{i+1}^\alpha$
- R-matrix operator : $R_{ij}(u) \ : V_i \otimes V_j \to V_i \otimes V_j$
- Yang-Baxter equation : $\underline{R_{ij}(u-v)R_{ik}(u)R_{jk}(v) = R_{jk}(v)R_{ik}(u)R_{ij}(u-v)}$
- Regularity : $\underline{R_{ij}(0) = P_{ij}}$
- Monodromy matrix : $\mathcal{T}_a(u) = R_{a,L}(u)R_{a,L-1}(u)\dots R_{a,1}(u)$
- Transfer matrix : $T(u) = \mathrm{tr}_a(\mathcal{T}_a(u))$

- RTT relation : $R_{12}(u-v)\mathcal{T}_1(u)\mathcal{T}_2(v) = \mathcal{T}_2(v)\mathcal{T}_1(u)R_{12}(u-v)$

- Commutativity of transfer matrices : $[T(u), T(v)] = 0$

- Family of committing charges : $\log T(u) = \sum_{n=0}^{\infty} \mathbb{Q}_{n+1} \frac{u^n}{n!}$

$$\mathbb{Q}_{n+1} = \frac{d^n}{du^n} \log T(u)|_{u=0} = \frac{d^{n-1}}{du^{n-1}} \left(T^{-1}(u) \frac{d}{du} T(u) \right) \bigg|_{u=0}$$

- Hamiltonian density : $\underline{H_{i,i+1} = R^{-1}_{i,i+1}(0)} \frac{d}{du} \underline{R_{i,i+1}(u)|_{u=0} = P_{i,i+1}} \frac{d}{du} \underline{R_{i,i+1}(u)|_{u=0}}$

- One can impose further restrictions on the level of R-matrix or Hamiltonian, such as unitarity $R_{12}(u)R_{21}(-u) = g(u)$, crossing, hermicity, symmetries etc.

- Given a solution for the Yang-Baxter equation, one can generate a whole family of solutions by acting with the following transformations :

  - $(\Omega \otimes \Omega) R(u) (\Omega^{-1} \otimes \Omega^{-1})$, $\Omega \in Aut(V)$: $\mathbb{Q}_n \to (\otimes^L \Omega) \mathbb{Q}_n (\otimes^L \Omega^{-1})$
  - rescaling of spectral parameter $u \to cu, \ \forall c \in \mathbb{C}$: $\mathbb{Q}_n \to c^{n-1} \mathbb{Q}_n$
  - $R(u) \to f(u) R(u), \quad f(0) = 1$
  - $PR(u)P, \ R(u)^T, \ PR^T(u)P \quad \to \quad P\mathcal{H}P, P\mathcal{H}^T P, \mathcal{H}^T$

There is no general way to solve YB

- There is no general way to find integrable Hamiltonian/solve YB. Roughly speaking there are three approaches :

  i. assume an symmetry (Lie algebra, Yangian, etc) and use it

  ii. directly solve functional or related dif equation R.S.Vieira '17'19

  Drinfeld, Faddeev, Reshetikhin, Kulish,..

- iii. use boost operator $\mathcal{B} = \sum_{a=-\infty}^{\infty} a H_{a,a+1}$ to generate higher charges $\mathbb{Q}_{r+1} = [\mathcal{B}, \mathbb{Q}_r]$

  impose commutativity $[\mathbb{Q}_i, \mathbb{Q}_j] = 0$ and solve the resulting algebraic eqns.

  Marius de Leeuw et al '19'20'20 Watch Marius talk at LIJC for details

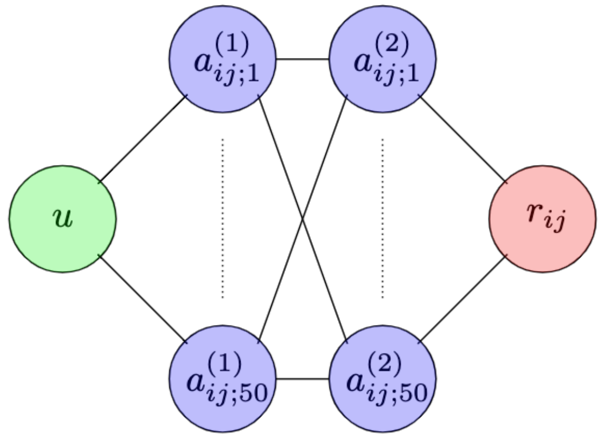
  De Leeuw, Pribytok, Ryan '19 classified all integrable 2d spin chains of difference form. They found 14 different classes in total, 8 of XYZ type and 6 of non-XYZ type :

$$H_{\mathrm{XYZ\,type}} = \begin{pmatrix} a_1 & 0 & 0 & d_1 \\ 0 & b_1 & c_1 & 0 \\ 0 & c_2 & b_2 & 0 \\ d_2 & 0 & 0 & a_2 \end{pmatrix}, \qquad H_{\mathrm{non-XYZ\,type}} = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ 0 & b_1 & b_3 & b_3 \\ 0 & c_1 & c_2 & c_3 \\ 0 & 0 & 0 & d_1 \end{pmatrix}$$

Machine Learning the R-Matrix

- We restrict the spectral parameter u to the interval (-1,1) and approximate each of real and imaginary parts of $R_{ij}(u)$ by NN with two hidden layers consisting of 50 neurons :

$\mathcal{R}_{ij}\left(u\right) = a_{ij}\left(u\right) + i\ b_{ij}\left(u\right)$:



; $r_{ij} = \{a_{ij}, b_{ij}\}$.

- YB loss function :

$$\mathcal{L}_{YBE} = \sum_{u,v\in(-1,1)} ||\mathcal{R}_{12}(u-v)\mathcal{R}_{13}(u)\mathcal{R}_{23}(v) - \mathcal{R}_{23}(v)\mathcal{R}_{13}(u)\mathcal{R}_{12}(u-v)||$$

where $||A|| = \sum_{\alpha,\beta=1}^{n} |A_{\alpha\beta}|$ and u and v run over 20000 random points in (-1,1)

- Regularity : $\mathcal{L}_{reg} = ||\mathcal{R}\left(0\right) - P||$

Hamiltonian : $\mathcal{L}_{H} = ||P\frac{d}{du}\mathcal{R}(u)|_{u=0} - H||$

Hermicity : $\mathcal{L}_{\dagger} = ||H - H^{\dagger}||$

- Total loss function :

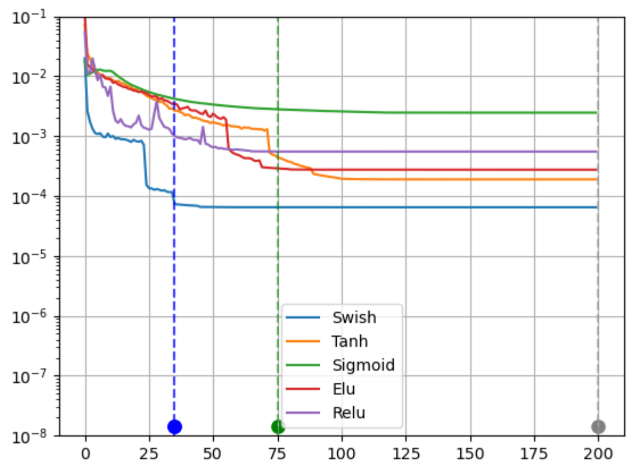
$$\mathcal{L} = \mathcal{L}_{YBE} + \mathcal{L}_{reg} + \lambda_H \mathcal{L}_H + \lambda_\dagger \mathcal{L}_\dagger + \ldots$$

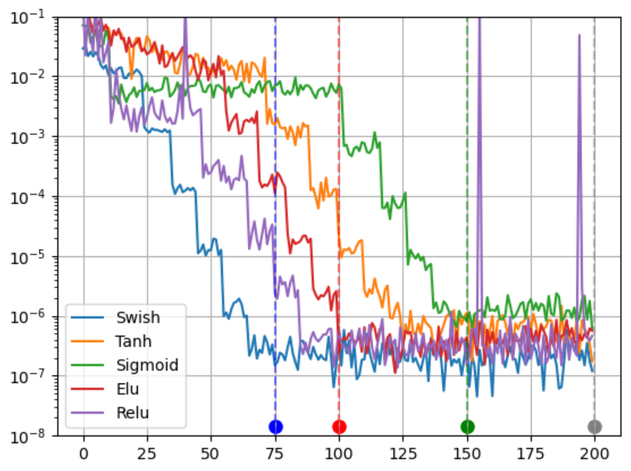
- Activation function : $swish(z) = \dfrac{z}{1+e^{-z}}$

- Optimization is done using Adam : $\beta_1 = 0.9,\ \beta_2 = 0.999$, $\eta = 10^{-3}, \ldots, 10^{-8},$ with step $= 10^{-1}$

- Comparison of activation functions :

Activation	Final Yang-Baxter Loss	Final Hamiltonian Loss	Saturation Epoch
sigmoid	2.5×10^{-3}	6.12×10^{-7}	150
tanh	1.90×10^{-4}	5.25×10^{-7}	125
$\Rightarrow$ swish	6.49×10^{-5}	1.51×10^{-7}	75
elu	2.75×10^{-4}	5.63×10^{-7}	100
relu	5.52×10^{-4}	4.63×10^{-7}	100





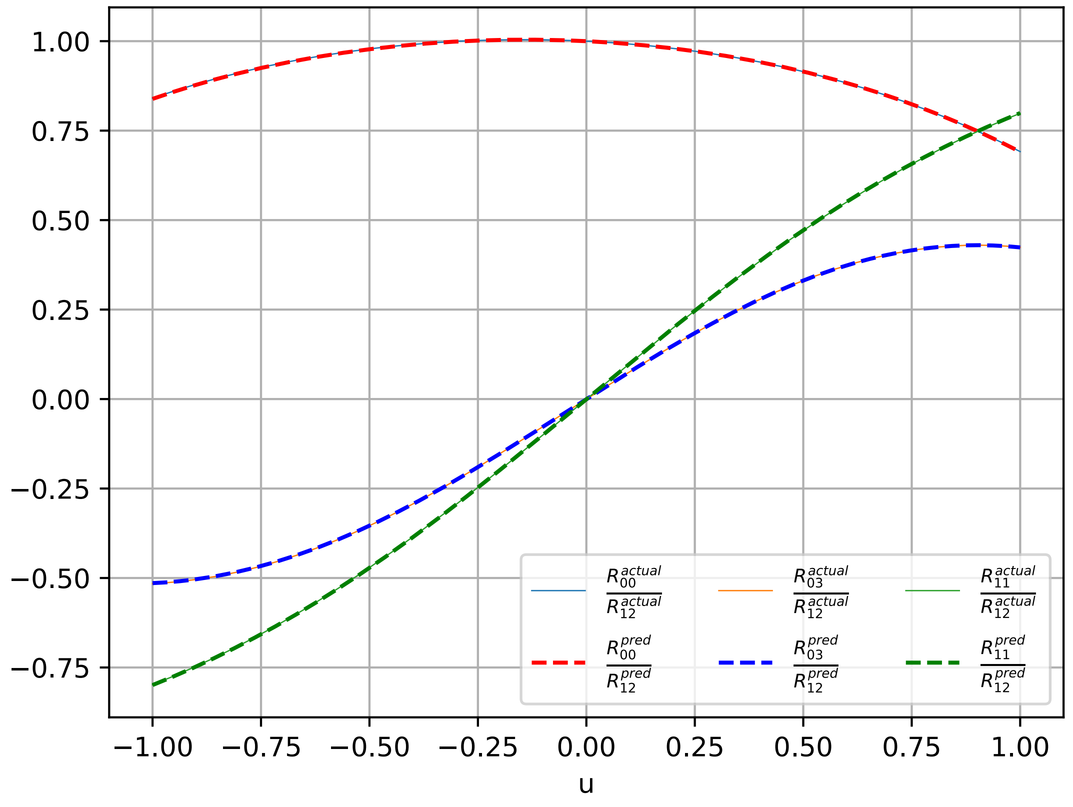
Hermitian XYZ model

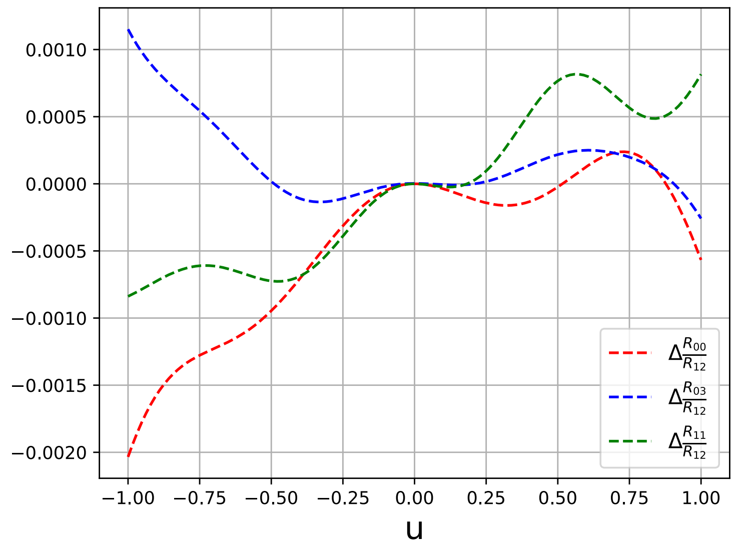
$$H_{XYZ}(J_x, J_y, J_z) = J_x S_1^x S_2^x + J_y S_1^y S_2^y + J_z S_1^z S_2^z = \begin{pmatrix} J_z & 0 & 0 & J_x - J_y \\ 0 & -J_z & 2 & 0 \\ 0 & 2 & -J_z & 0 \\ J_x - J_y & 0 & 0 & J_z \end{pmatrix}$$

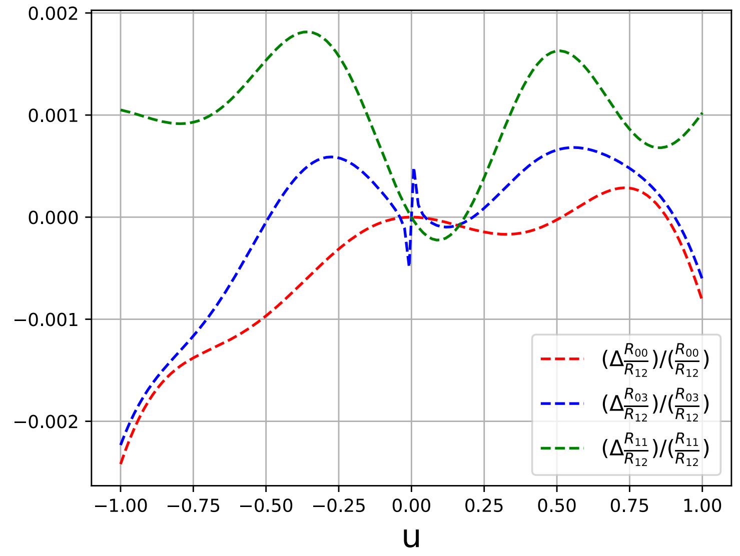
$$J_x = 1 + \sqrt{m}\ \mathrm{sn}(2\eta|m)/2$$
$$J_y = 1 - \sqrt{m}\ \mathrm{sn}(2\eta|m)/2$$
$$J_z = \mathrm{cn}(2\eta|m)\mathrm{dn}(2\eta|m)$$

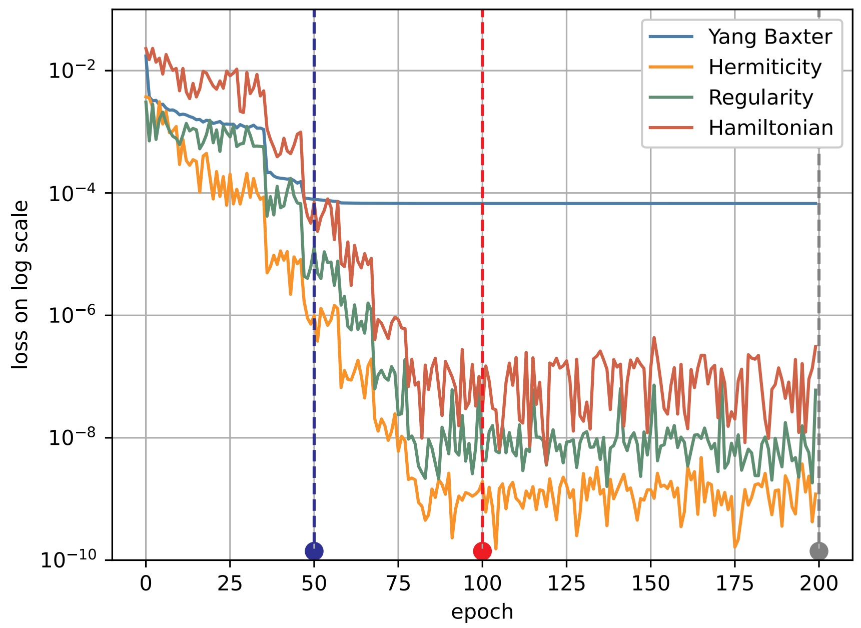
$$R(u) = \begin{pmatrix} a & 0 & 0 & d \\ 0 & b & c & 0 \\ 0 & c & b & 0 \\ d & 0 & 0 & a \end{pmatrix}$$

, where $a(u),\ b(u),\ c(u),\ d(u)$ are given in terms of Jacobi elliptic functions









$\eta = \pi/3, \quad m = 0.6$

$$H_{class-1} = \begin{pmatrix} 0 & a_1 & a_2 & 0 \\ 0 & a_5 & 0 & a_3 \\ 0 & 0 & -a_5 & a_4 \\ 0 & 0 & 0 & 0 \end{pmatrix} \qquad R_{\text{class}-1}(u) = \begin{pmatrix} 1 & \frac{a_1(e^{a_5 u}-1)}{a_5} & \frac{a_2(e^{a_5 u}-1)}{a_5} & \frac{(a_1 a_3 + a_2 a_4)}{a_5^2}(\cosh(a_5 u) - 1) \\ 0 & 0 & e^{-a_5 u} & \frac{a_4(1-e^{-a_5 u})}{a_5} \\ 0 & e^{a_5 u} & 0 & \frac{a_3(1-e^{-a_5 u})}{a_5} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$a_1 a_3 = a_2 a_4$$

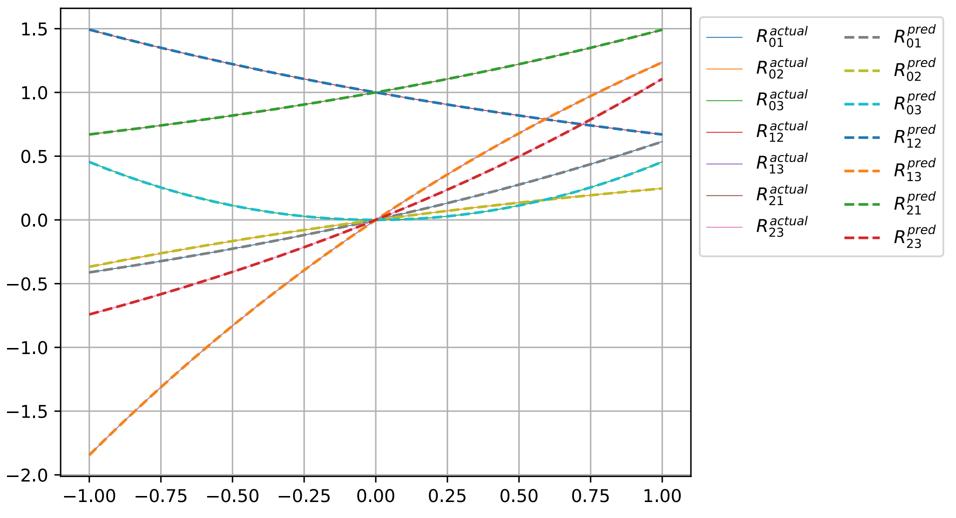
$a_1 = 0.5$

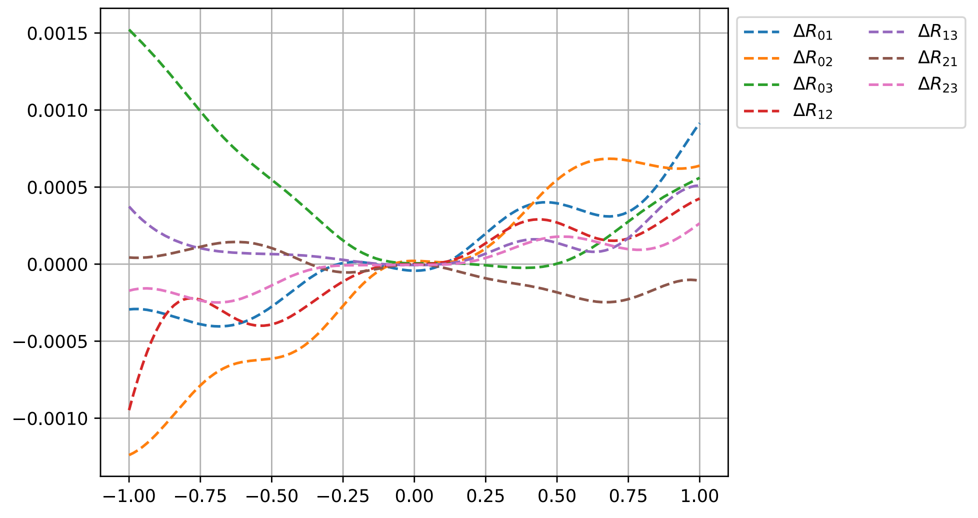
$a_2 = 0.3$

$a_3 = 0.9$

$a_4 = 1.5$

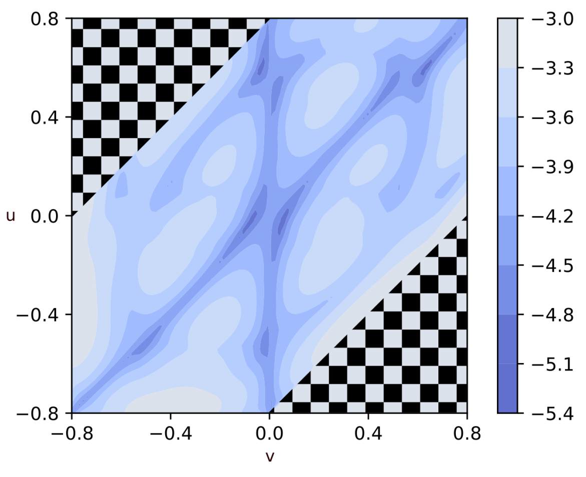
$a_5 = 0.4$

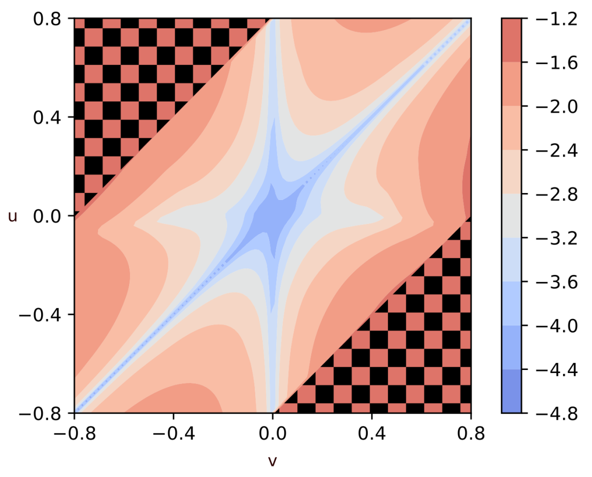


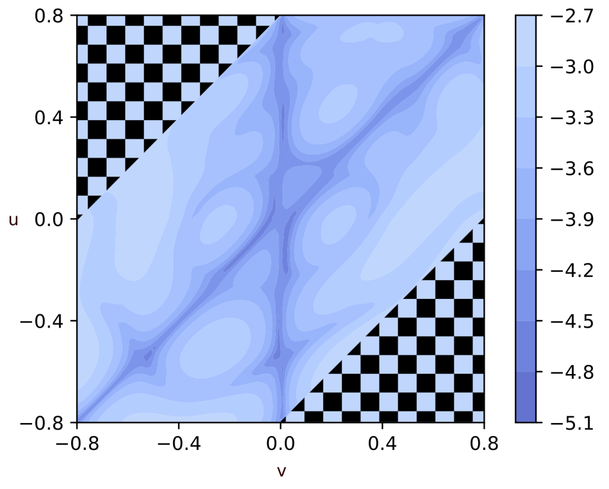


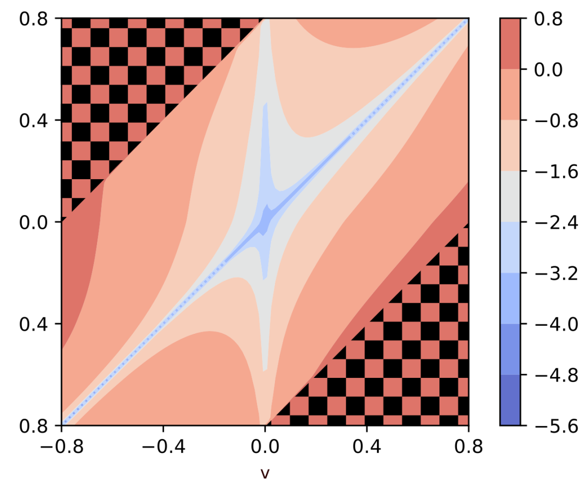
Similar $10^{-3} - 10^{-4}$ precision for all 14 classes

Without exact solution it is reasonable to define normalised YB :
$$\tilde{\mathcal{L}} = \frac{||\mathcal{R}_{12}(u-v)\mathcal{R}_{13}(u)\mathcal{R}_{23}(v) - \mathcal{R}_{23}(v)\mathcal{R}_{13}(u)\mathcal{R}_{12}(u-v)||}{||\mathcal{R}_{12}(u-v)\mathcal{R}_{13}(u)\mathcal{R}_{23}(v)||}$$


$H_{6v,1}$


$H_{6v,1}\quad non-int\ deform$


$H_{class-4}$

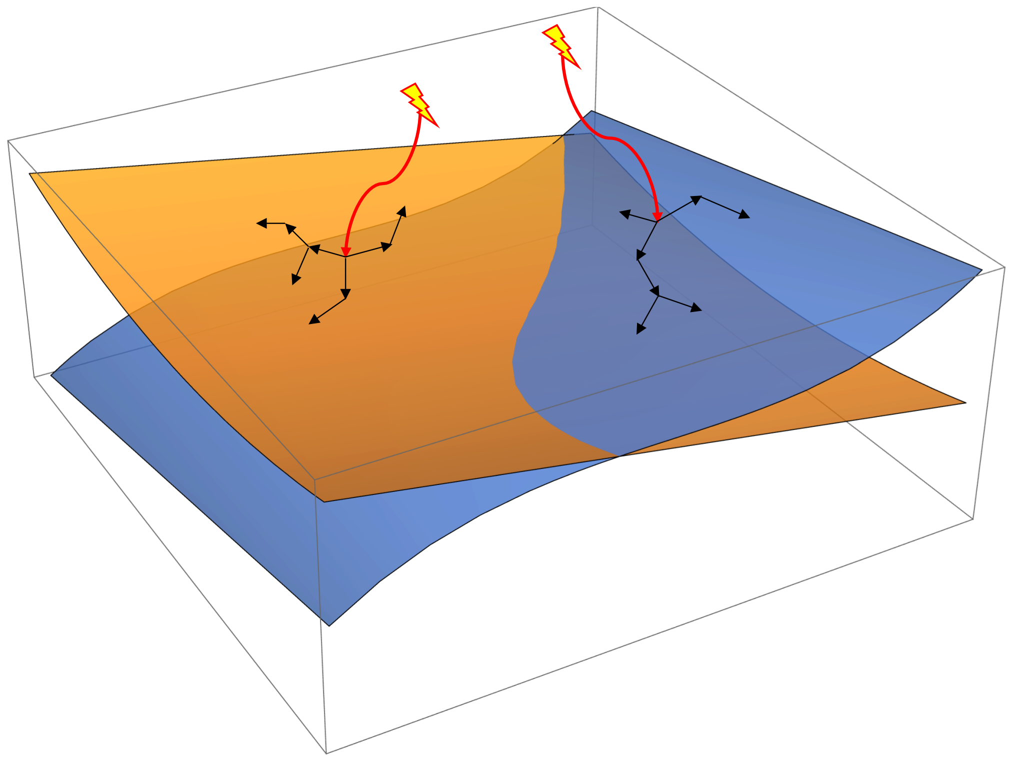

$H_{class-4}\quad non-int\ deform$

Explorer

- In explorer mode we i) don't provide a certain integrable Hamiltonian, instead we ask the NN to find an example of integrable model. It can be either general search without any restrictions at all or we can narrow the search to a certain class of hamiltonians imposing symmetry on the level of R-matrix or Hamiltonian, specifying ansatz etc. ii) Secondly we use *warm-start initialisation* and *repulsion* to find new integrable models in the vicinity of the already learnt one :

$$\mathcal{L}_{repulsion} = \exp\left(-||H - H_o||/\sigma\right)$$

- Explorer

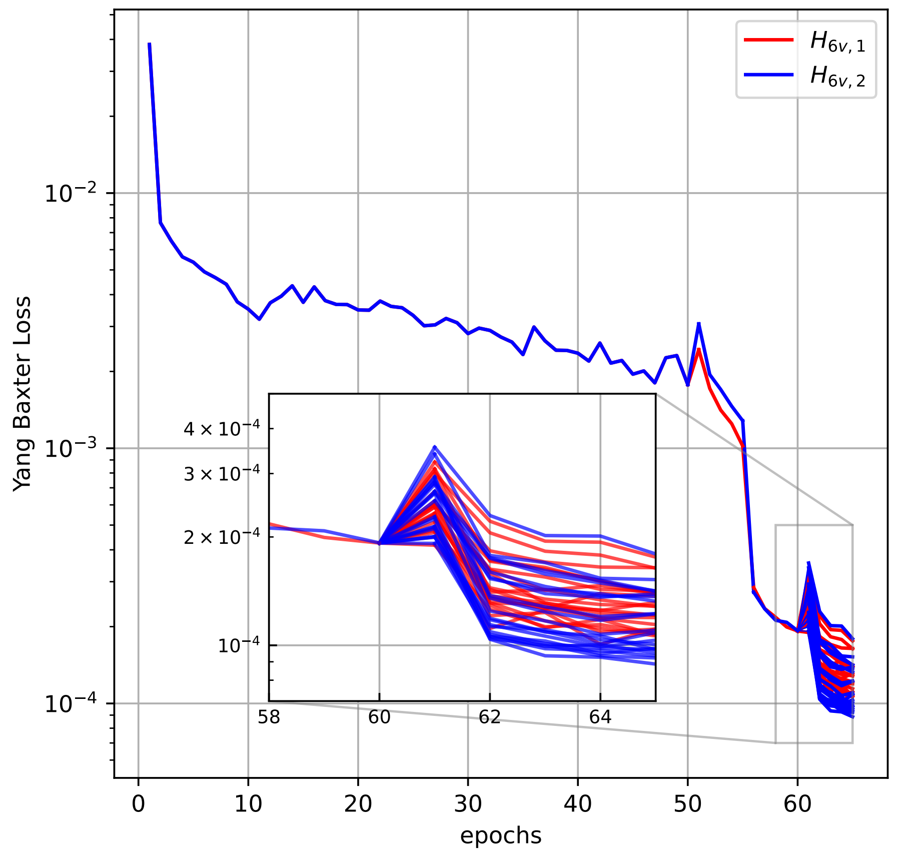


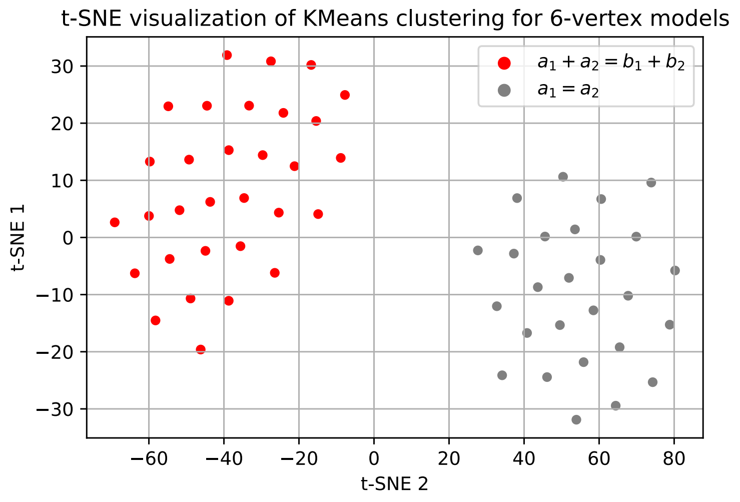
Toy model. 6-vertex

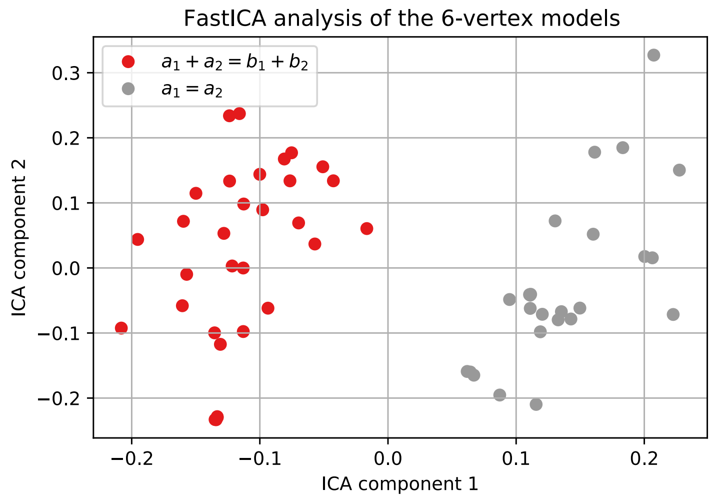
- Six-vertex models :

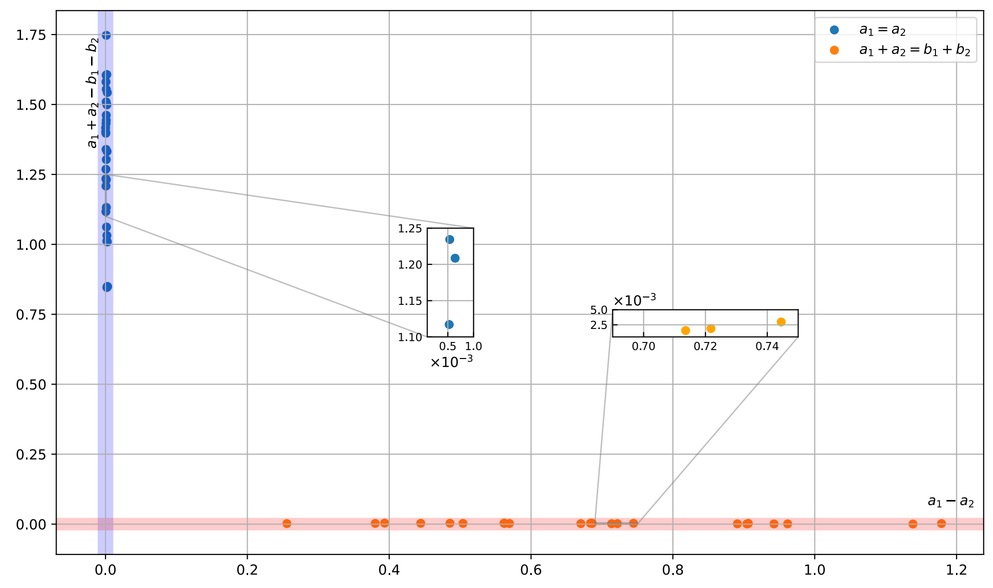
$$H_{6\mathrm{v}} = \begin{pmatrix} a_1 & 0 & 0 & 0 \\ 0 & b_1 & c_1 & 0 \\ 0 & c_2 & b_2 & 0 \\ 0 & 0 & 0 & a_2 \end{pmatrix} \qquad \begin{aligned} H_{6\mathrm{v},1} &: \quad a_1 = a_2 \\ H_{6\mathrm{v},2} &: \quad a_1 + a_2 = b_1 + b_2 \end{aligned}$$

- Explorer found 50 Hamiltonians and then clusterised in two families corresponding to two classes of six-vertex models :





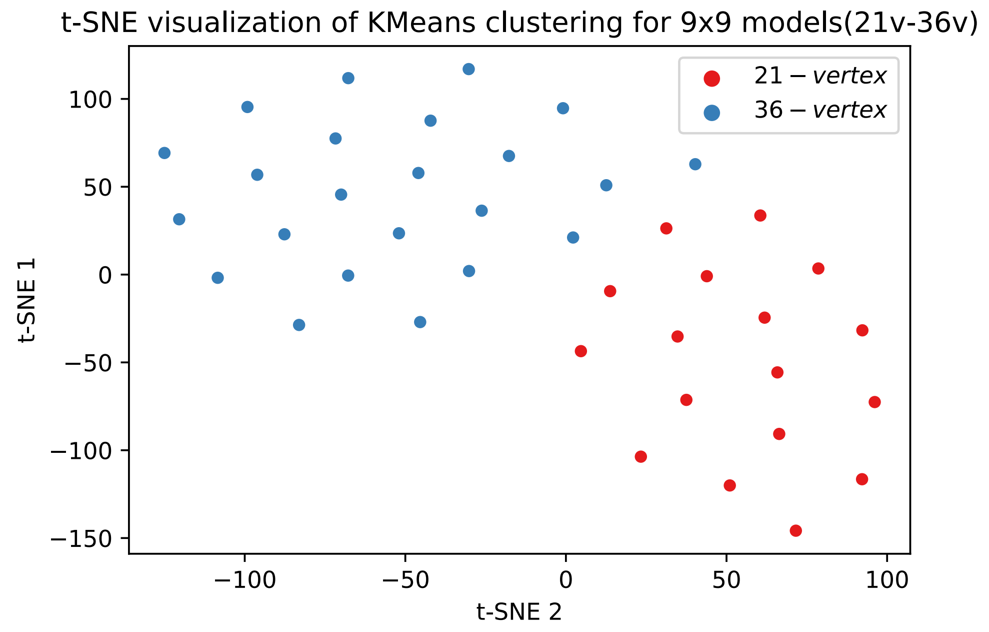




First look into 3d

- We just started to ran random initialisations and see many potentially new integrable Hamiltonians with various number of vertices : 15, 20, 21, 24, 27, 33, 36, …

- One of the very first runs and clusterisation for 21 and 36 vertex models :



- Hopefully the first version of the map in a few weeks

Conclusions and future directions

- We built the NN which can i) find the numerical R-matrix for a given integrable spin chain of difference form and ii) search for new integrable Hamiltonians.

- Carve out the map of all integrable spin-chains of difference form with three-dimensional site space. *in progress*
- R-matrices of nondifference form. Especially interesting that ones related to AdS/CFT. Also boundary YB, modifyed YB etc
- Is it helpful for analytics? At least two strategies: 1) symbolic regression 2) use the learnt numerical solution to detect the symmetries and then use them to find analytical solution
- "R-matrix Bootstrap" similarly to 2d S-matrix Bootstrap? Relax YB, use just unitarity, analyticity, hermicity, symmetries etc to find the solution? *in progress*
- Solve YB for S-matrices in 2d integrable QFTs. *in progress*
- NN as a tool to measure complexity of Hamiltonians? Landscape of complexity?

Thank you for your attention!