Physics-Informed Neural Networks: COMSOL Multiphysics[®] and MATLAB[®]



Martin Kožíšek kozisek@humusoft.cz



Jaroslav Jirkovský jirkovsky@humusoft.cz

Schedule

- 1. Introduction to Surrogate Models in COMSOL
- 2. Deep Neural Networks in COMSOL
 - Workflow
 - DNN Architecture
 - One Neuron
 - Activation Function
 - Analytical Expression For a Deep Neural Network Model
 - Influence of Network Architecture and Activation Function
- 3. Learning and Validation Process
- 4. Example in COMSOL
- 5. More Realistic Example in COMSOL
- 6. Introduction Neural Networks in MATLAB
- 7. Physics-Informed Neural Networks
- 8. Example in MATLAB

Introduction to Surrogate Models in COMSOL



<u>kozisek@humusoft.cz</u> <u>www.linkedin.com/in/martinkozisek/</u>



What Are Surrogate Models?

A simplified model that replaces a complex computational model (e.g., a simulation) without significant loss of accuracy.



Three (or Four?) Types of Surrogate Models in COMSOL Multiphysics

1. DNN Based (Deep Neural Network)

Capturing complex, nonlinear relationships in large sets of data (Digital Twins), included in COMSOL Multiphysics[®] without add-on products

2. GP Based (Gaussian Process)

A probabilistic framework that provides both predictions and uncertainty (Opitmization and Uncertainty Quantification), can handle only thousands of data tables, requires add-on modules.

3. PCE Based (Polynomial Chaos Expansion)

Representing the influence of random variables (Sensitivity Analysis), can handle only thousands of data tables, requires add-on modules.

4. (Other Methods: Model Order Reduction)



Deep Neural Networks in COMSOL Multiphysics



<u>kozisek@humusoft.cz</u> www.linkedin.com/in/martinkozisek/

Workflow

- Setting Up a full simulation model
- Computing for a "some" range of parameters (Design of Experiments)
 - Manual range (parametric sweep)
 - Latin hypercube sampling (covers the space of parameters)
- Definition and training in a new function type: Deep Neural Network
- Surrogate model: multidimensional function approximation
- Calling the Deep Neural Network function, e.g. in App or in Digital Twin



DNN Architecture

What does DNN look like, and how do you design it?

- Input layer: number of inputs
- A series of hidden layers: iterative process, knowledge, empirical testing, trial and error
- An output layer: number of outputs

Notes: Each layer consists of a number of neurons. Too few layers/neurons may lead to **underfitting**. Excess layers/neurons can cause **overfitting**.



The figure shows a graph for a network with three hidden layers, five input nodes, and two output nodes. COMSOL Multiphysics uses Dense feed-forward networks (Dense = every neuron is connected to every neuron in the adjacent layer, Feed-Forward = data flows in one direction from input to output).

The Neuron

Each neuron responds to n inputs by returning a scalar output.



Note: DNN Training is searching for optimal values of Biases and Weights



Activation Functions

Activation Functions in COMSOL Multiphysics 6.3:

- Linear: no activation, just a weighted sum. Outputs: (-∞, ∞)
- ReLU: returns max(0, x) zero or linear for x > 0. Outputs: [0, ∞),
- ELU: returns x for x > 0, or e^x pro x ≤ 0.
 It is smooth ReLU. Outputs: [0, ∞).
- Sigmoid: $\frac{1}{1+e^{-x}}$ with outputs: (0, 1)
- Tanh: $\frac{e^x e^{-x}}{e^x + e^{-x}}$ with outputs: (-1, 1)



Exponential Linear Unit (ELU)



Analytical Expression for DNN Example

Thermal Microactuator is a MEMS device that converts an electric input signal into mechanical motion.

Simulation model settings:

- Electric current conduction,
- Heat conduction with Joule heating
- Thermal expansion (Stress and Strain)

Parameters:

- Actuator length
- Applied voltage

Model computes maximum displacement



Analytical Expression for DNN Example

Thermal Microactuator is a MEMS device that converts an electric input signal into mechanical motion.

Simulation model settings:

- Electric current conduction,
- Heat conduction with Joule heating
- Thermal expansion (Stress and Strain)

Parameters:

- Actuator length
- Applied voltage

Model computes maximum displacement



Analytical Expression for DNN Example

Thermal Microactuator is a MEMS device that converts an electric input signal into mechanical motion.

Simulation model settings:

- Electric current conduction,
- Heat conduction with Joule heating
- Thermal expansion (Stress and Strain)

Parameters:

- Actuator length
- Applied voltage

Model computes maximum displacement



dnn(x1,x2) =

 $\begin{aligned} & tanh(w3_11*(tanh(w2_11*(tanh(w1_11*(x1)+w1_21*(x2)+b1_1))+w2_21*(tanh(w1_12*(x1)+w1_22*(x2)+b1_2))+w2_31*(tanh(w1_13*(x1)+w1_23*(x2)+b1_3))+w2_41*(tanh(w1_14*(x1)+w1_24*(x2)+b1_4))+b2_1))+w3_21*(tanh(w2_12*(tanh(w1_11*(x1)+w1_21*(x2)+b1_1))+w2_22*(tanh(w1_12*(x1)+w1_22*(x2)+b1_2))+w2_32*(tanh(w1_13*(x1)+w1_23*(x2)+b1_3))+w2_42*(tanh(w1_14*(x1)+w1_24*(x2)+b1_4))+b2_2))+w3_1)\end{aligned}$



The Table shows optimized values of weights and biases. The data needs to be scaled before the network can be optimized, since the network uses the tanh activation function, and the output values of tanh are limited to (-1, 1).

Influence of Network Architecture and Activation Function

For regression tasks, as an alternative to using the tanh activation function, a combination of the ReLU and Linear activation functions can be used. Comparision:

- Tanh [2,4,8,4,1]
- ReLU + Linear [2,4,8,4,1] Loss 0.2
- ReLU + Linear [2,8,16,8,1]
- ReLU + Linear [2,16,32,16,1]
- ReLU + Linear [2,32,64,32,1]







Training and Validation Process



<u>kozisek@humusoft.cz</u> <u>www.linkedin.com/in/martinkozisek/</u>

Training a DNN

- Training involves optimizing weights and biases to minimize error.
- Objective of training: Align the surrogate model closely with the finite element model.
- COMSOL uses the ADAM (Adaptive Moment Estimation) algorithm, which evaluates the first (mean) and second (variance) moments of the gradient of the loss function to optimize weights and biases.
- Error measurement is calculated via the loss function.
- Different types of loss functions can be used.
- The default loss function is *Root-mean-square error* (RMSE).



ADAM algorithm is adaptive stochastic gradient descent method (SGD). The stochastic method randomly selects one small mini-batch of data ata time, to perform an update on the network parameters. The randomness in selecting data points introduces variability in the gradient estimates, which helps escaping local minima in the loss landscape, potentially leading to better generalization on unseen data. As a result, the updates to the network parameters are noisy, as can be seen from the convergence plots.

Training and Validation Loss

Training data are split into Primary training data and Validation data.

- Training loss:
 - Reflects model performance on the primary training data.
- Validation loss:
 - Indicates model performance on a separate, unseen subset of data. Why?
 If we train and test the model on the same data, it can learn the specifics of that data, leading to poor generalization.

Note: When I was a child, I loved solving equations. Every math book had a few step-by-step examples, and I learned from these examples. Then, I solved new problems using what I had learned and checked my answers with the correct solutions—that's like training loss and validation loss.



Training and Validation Settings

- Learning rate: the step size during the optimization process. Too small rate can lead to the model getting stuck in a local minimum. Too large rate can result in overshooting the minimum and poor convergence.
- Batch size: the division of training data into subsets during the optimization process. Too small batch can lead to noisy gradient updates and longer training times. Too large batch might lead to poor generalization.
- Number of epochs: number of complete passes through the entire dataset. Too few epochs can result in underfitting, where the model has not adequately learned from the training data. Too many epochs can lead to overfitting, where the model learns the noise in the training data and performs poorly on new, unseen data.
- *Validation data fraction:* the size of the validation data sample.

 Training and validation 						
Method:	Adam 🔹					
Learning rate:	1e-3					
Weight decay:	0					
Batch size:	512					
Loss function:	Root-mean-square error 🔹					
Random seed type:	Fixed 🔹					
Random seed:	0					
- Stop condition						
Number of epochs:	50000					
Validation data						
Validation data:	Random sample of data values 🔹					
Validation data fraction:	0.1					
Random seed type:	Fixed 🔹					
Random seed:	0					



Visualization of an overfitted model (left) and Well-fitted model (right)



0.5

-0.5

0

0.5

0 0.10.0

-0.0 -0.1



Martin Kožíšek

1D Burger's equation

- Equation: $\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} \frac{0.01}{\pi} \frac{\partial^2 u}{\partial x^2} = 0$
- Space and time domain: $x \in \langle -1, 1 \rangle$, $t \in \langle 0, 1 \rangle$
- Initial conditions: $u(x, 0) = -sin(\pi x)$
- Boundary conditions: u(-1,t) = 0, u(1,t) = 0





How to Type Your Own Equation in COMSOL

- 1D Component
 - Interval
- ID General Form PDE
 - Filling the equation into the template
 - Initial Condition Settings
 - Dirichlet Boundary Condition
- User Controled Mesh
 - Edge with Distribution
 - 1000 elements
- Time Dependent Study
 - Output times: range(0,1/50,1)
 - Adaptive mesh refinement

Model Builder	+				
← → ↑ ↓ ◙ ≣↑ ♥ ≣↓ ♥ ≣∎ ♥ ♥ ♥ Type filter text	C				
 ✓ Surgers_Equation_COMSOL_jemnejsi.mph (rational second se	Doot) Settings General Form PDE Label: General Form PDE 1 Domain Selection Selection: All domains 1	Settings Time Dependent = Compute			
Dirichlet Boundary Condition 1		Label: Time Dependent			
at ⁼ Equation View	> Override and Contribution	× Study Settings			
 Mesh 1 Size Edge 1 Distribution 1 Mesh 2 Mesh 3 Mesh 4 Mesh 5 Mesh 6 Mesh 7 Mesh 8 Mesh 9 Mesh 10 Mesh 11 Study 1 Step 1: Time Dependent Step 1: Time Dependent 	 ✓ Equation Show equation assuming: Study 1, Time Dependent $e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = f$ $\nabla = \frac{\partial}{\partial x}$ ✓ Conservative Flux Γ -ux*0.01/pi ✓ Source Term <i>f</i> -u*ux ✓ Damping or Mass Coefficient d_a 1 	Output times: range(0,1/50,1) Tolerance: Physics controlled > Results While Solving > Physics and Variables Selection > Values of Dependent Variables > Store in Output > Mesh Selection ~ Adaptation Adaptive mesh refinement: Adaptive mesh refinement: Adaptation in geometry: Geometry 1 Time-interval control Time-interval length: Manual			
	 Mass Coefficient 	Adaptation method: Longest edge refinement			
	e _a 0	Geometric Entity Selection for Adaptation			
		> Study Extensions			

How to Type Your Own Equation in COMSOL

- 1D Component
 - Interval
- ID General Form PDE
 - Filling the equation into the template
 - Initial Condition Settings
 - Dirichlet Boundary Condition
- User Controled Mesh
 - Edge with Distribution
 - 1000 elements
- Time Dependent Study
 - Output times: range(0,1/50,1)
 - Adaptive mesh refinement

Model Builder	+				
← → ↑ ↓ ◙ ≣↑ ♥ ≣↓ ♥ ≣∎ ♥ ♥ ♥ Type filter text	C				
 ✓ Surgers_Equation_COMSOL_jemnejsi.mph (rational second se	Doot) Settings General Form PDE Label: General Form PDE 1 Domain Selection Selection: All domains 1	Settings Time Dependent = Compute			
Dirichlet Boundary Condition 1		Label: Time Dependent			
at ⁼ Equation View	> Override and Contribution	× Study Settings			
 Mesh 1 Size Edge 1 Distribution 1 Mesh 2 Mesh 3 Mesh 4 Mesh 5 Mesh 6 Mesh 7 Mesh 8 Mesh 9 Mesh 10 Mesh 11 Study 1 Step 1: Time Dependent Step 1: Time Dependent 	 ✓ Equation Show equation assuming: Study 1, Time Dependent $e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = f$ $\nabla = \frac{\partial}{\partial x}$ ✓ Conservative Flux Γ -ux*0.01/pi ✓ Source Term <i>f</i> -u*ux ✓ Damping or Mass Coefficient d_a 1 	Output times: range(0,1/50,1) Tolerance: Physics controlled > Results While Solving > Physics and Variables Selection > Values of Dependent Variables > Store in Output > Mesh Selection ~ Adaptation Adaptive mesh refinement: Adaptive mesh refinement: Adaptation in geometry: Geometry 1 Time-interval control Time-interval length: Manual			
	 Mass Coefficient 	Adaptation method: Longest edge refinement			
	e _a 0	Geometric Entity Selection for Adaptation			
		> Study Extensions			

- Export Data
- Functions: Deep Neural Network 1
 - Data: exported *.txt table
 - Data Columns:
 arguments: x, t
 function value: u
 - Layers and Activations:[2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation: Default settings + Train on GPU First 10000 epochs with rate 1e-3 Other 5000 epochs with rate 1e-4 Continue Training
- Plot DNN in the x-t-u space

	≣↓ ▼	Seti Globa	t ing I Defi	S nitior	15
со эг 1	Pi	L.mph (root) Parameters			
cc	9=	Variables Equation Contributions	•		
		Functions	•	f⊗) Q	Analytic
		Geometry Parts Mesh Parts	•	\sim	Interpolation Piecewise
u m	*	Default Model Inputs Materials	∧ ∕~	Gaussian Pulse Ramp	
u		Load and Constraint Groups Thermodynamics	۲ ۲	л Г	Rectangle Step
		Parameter Estimation Extra Dimensions	۲ ۲	~	Iriangle Waveform
	ei	Show More Options	MW	Random	
		Node Group		t _c	External
	Ē	Group by Type		t,	MATLAB
	?	Help	F1	9	Elevation (DEM)
				A.	Image Least-Squares Fit
					Gaussian Process
			$\overline{1}$	1. 1. j.	Polynomial Chaos Expansion
			L	\sim	Deep Neural Network
				\sim	Partial Fraction Fit
				$\hat{\mathbb{N}}$	Function Switch

Baking a Deep Neural Network (DNN)

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table (2)
 - Data Columns: arguments: x, t function value: *u*
 - Layers and Activations: [2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation: Default settings + Train on GPU First 10000 epochs with rate 1e-3 Other 5000 epochs with rate 1e-4 Continue Training
- Plot DNN in the x-t-u space

:≣↓	• III • ≜ • • C	Setting: Global Defir	S nitions	Setting Deep Neura Plot B	S al Network Create Plot	i ^{ro} Train Model 📩 Continu work 1	e Training	
COM pr	SOL.mph (root)			 Layers 				
1 a	 Parameters Variables Equation Contributions 	,		► Type	Settings	put features=2		
	Functions	•	^f ∞ Analytic	Dense	✓ Hidden.	' Output features=8. Activatic	n=tanh	
	Geometry Parts	•	Λ . Interpolation	Dense	✓ Hidden,	Output features=16, Activati	on=tanh	
u	Mesh Parts	•	∧ Piecewise	Dense	✓ Hidden,	Hidden, Output features=32, Activation= Hidden, Output features=8, Activation=t		
	Default Model Inputs		✓ Gaussian Pulse	Dense	✓ Hidden,			
n u	Materials	•	Ramp	Dense	 Output, 	Output features=1, Activatio	n=tanh	
	Load and Constraint Group Thermodynamics Parameter Estimation Extra Dimensions Show More Options Node Group Group by Type Help	inics , imation , ons , pptions // e F1	Step Step Triangle Waveform Normal Distribution Random External MATLAB Elevation (DEM) Least-Squares Fit Gaussian Process Polynomial Chaos Exp Deep Neural Network	 ↑ ↓ + ⇒ ↓ Output features: Activation: tanh Layer configuration: [2,8,16,32,8,1] ✓ Data Data source: File Decimal separator: Point USERS\Desktop\Data.txt Filename: 2 ➡ Browse ▼ ➡ Import C Refresh ✓ Ignore NaN/Inf data points 			- - - -	
				 Data Co 	lumn Setti	ngs		
			M Function Switch	Column:	Туре	Settings		
				Х	Argu 🔻	Name=x1, Scaling=to01		
				t	Argu 🔻	Name=x2, Scaling=to01		

Func

 Name=dnn1, Scaling=to01

Ē

Baking a Deep Neural Network (DNN)

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table
 - Data Columns: arguments: x, t 3 function value: *u*
 - Layers and Activations: [2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation: Default settings + Train on GPU First 10000 epochs with rate 1e-3 Other 5000 epochs with rate 1e-4 Continue Training
- Plot DNN in the x-t-u space

	≣† ≁	•□‡ 	Settings Global Defir	S nitions	Setting Deep Neura Plot R	S al Network Create Plot	[[] Train Model [述 Continue work 1	Training
r 1	MSO Pi	L.mph <i>(root)</i> Parameters			✓ Layers			
- 	a=	Variables			₩ Туре	Settings		
		Functions	۲ ۲	^f ∞ Analytic	Input Dense	 Input, Input, Inp	out features=2 Output features=8. Activation	n=tanh
a)		Geometry Parts Mesh Parts	۲ ۲	 ∧ Interpolation ∧ Piecewise ∧ Gaussian Pulse 	Dense Dense	 Hidden, Hidden, Hidden, 	Output features=16, Activatio Output features=32, Activatio	n=tanh
r J	*	Default Model Inputs Materials Load and Constraint Group Thermodynamics	s •	Ramp Rectangle Step Triangle	Dense	Hidden, Output,	Output features=8, Activation Dutput features=1, Activation	ı=tanh ı=tanh
_	ei	Parameter Estimation Extra Dimensions Show More Options	•	Waveform Normal Distribution Random	Output featu Activation: Layer configu	ures: 1 ta uration: [2,8	anh ,16,32,8,1]	•
	2	Node Group Group by Type Help	F1	 External MATLAB Elevation (DEM) Image 	 Data Data source: Decimal sepa 	File arator: Po	e int RS\Deskton\Data txt	•
				 Least-Squares Fit Gaussian Process Polynomial Chaos Exp Deep Neural Network 	Filename:	■ E E C F aN/Inf data p	irowse ▼ ा Import Refresh points	
				Partial Fraction Fit Eunction Switch	✓ Data Co	lumn Settii	ngs	
					Column: X	Type Argu 🔻	Settings Name=x1, Scaling=to01	
					t	Argu 🔻	Name=x2, Scaling=to01	

Func
Vame=dnn1, Scaling=to01

Ē

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table
 - Data Columns: arguments: x, t function value: u
 - Layers and Activations:
 [2, 8, 16, 32, 8, 1] with *tanh* activation 4
 - Training and Validation:
 Default settings + Train on GPU
 First 10000 epochs with rate 1e-3
 Other 5000 epochs with rate 1e-4
 Continue Training
- Plot DNN in the x-t-u space

	∎↓ ▼ MSO	- □ ↓ ■ • • • • L.mph (root)	Setting: Global Defin	S hitior	ns	Setting Deep Neura Image: Deep Deep Deep	S Al Netw Create F O Neura	/ork 기ot I Net	f≪ Train M work 1
r 1	Pi	Parameters				✓ Layers			
	a=	Variables				•• Туре	Set	tings	
20		Equation Contributions	•			Input	• Inp	ut, In	put feature
		Functions	•	f(X) Q	Analytic	Dense	▼ Hid	den,	Output fea
		Geometry Parts	•	Λ.	Interpolation	Dense	Hid Hid	den,	Output fea
		Mesh Parts	•	\sim	Piecewise	Dense	Hid H Hid H Hid H Hid H	den,	Output fea
	<	Default Model Inputs		Л	Gaussian Pulse	Dense	▼ Hid	den.	' Output fea
n		Materials	•	_	Ramp	Dense	▼ Out	put (Output fea
u		Load and Constraint Group	s ▶		Rectangle				
		Thermodynamics	•		Step	↑↓+≣\			
		Parameter Estimation	•	Triangle		Triangle	Output featu		
		Extra Dimensions		•	Â.	Waveform	Activation:		ti
	θ	Show More Options			Normal Distribution	Layer configu	uration:	[2,8	8,16,32,8,1]
	t :	Node Group		/////	External	Data			
	畫	Group by Type		TE	MATLAR	✓ Data			
	?	Help	F1		Elevation (DEM)	Data source:		Fil	e
						Decimal sepa	arator:	Po	int
_				- 	Least-Squares Fit			USE	RS\Deskto
					Gaussian Process	Filename:		🗁 E	Browse 🔻
				بنې:	Polynomial Chaos Exp	_		C' F	Refresh
				$\overline{\mathbf{n}}$	Deep Neural Network	✓ Ignore Na	aN/Inf c	lata p	points
					Partial Fraction Fit	✓ Data Co	lumn S	Setti	ngs
				$(\hat{\mathbb{N}})$	Function Switch	>>	_		
						Column	Туре		Settings
						X	Argu	•	Name=x
						t	Argu	•	Name=x2
						u	Fund	: •	Name=d

		∇	
rk			
ot 🕅 Train	M	odel 🔯 Continue Training	
letwork 1		Ę	
igs			
Input feat	ure	s=2	
n, Output	feat	tures=8, Activation=tanh	
n, Output	feat	tures=16, Activation=tanh	
n, Output	feat	tures=32, Activation=tanh	1
n, Output	feat	tures=8, Activation=tanh	1
ıt, Output f	feat	ures=1, Activation=tanh	1
			1
1 tanh	4)	
2,8,16,32,8	,1]		
		 Training and Valida 	ation
		Method:	Adam
File		Learning rate:	1e-3
Point		Weight decay:	0
SERS\Desk	toŗ	Batch size:	512
Browse Refresh	•	Loss function:	Root-mean-square error
a points		Random seed type:	Fixed
ttings		Random seed:	0
congo		✓ Train on GPU	
Setting	gs	- Stop condition	
 Name= 	=x1	Number of epochs:	1000
▼ Name=	=x2	Validation data	
▼ Name=	=dr	Validation data:	Random sample of data values
		Validation data fraction:	0.1
		Random seed type:	Fixed
		Random seed:	0

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table
 - Data Columns:
 arguments: x, t
 function value: u
 - Layers and Activations:[2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation:
 Default settings + Train on GPU
 First 10000 epochs with rate 1e-3
 Other 5000 epochs with rate 1e-4
 Continue Training
- Plot DNN in the x-t-u space

	 ▼□ ▼ Setting Global Def OL.mph (root) 	JS initior	15	Setting Deep Neur Plot a Label: Dee
or 1 a=	Parameters Variables Equation Contributions	f∞	Analytic	 Layers Type Input
u m u	Geometry Parts Mesh Parts Default Model Inputs Materials Load and Constraint Groups Thermodynamics	へ へ へ て い く	Interpolation Piecewise Gaussian Pulse Ramp Rectangle Step Triangle	Dense Dense Dense Dense Dense
s (((([ff] ei	Parameter Estimation Extra Dimensions Show More Options Node Group Group by Type Holp E1		Waveform Normal Distribution Random External MATLAB	Output feature Activation: Layer config
		>>>	Elevation (DEM) Image Least-Squares Fit Gaussian Process Polynomial Chaos Exp Deep Neural Network	Decimal sep Filename: Ignore N V Data Co
		∬C M	Partial Fraction Fit	Column: X t u

Setting	S			~				
Deep Neura	al Netv	vork						
🖲 Plot 🐻	🛛 Plot 🐻 Create Plot 🚾 Train Model 🔛 Continue Training							
abel: Dee	o Neura	al Net	work 1					
 Layers 								
Туре	Set	ttings						
Input	▼ Inp	ut, In	put features	5=2				
Dense	▼ Hic	lden,	Output feat	ures=8, Activation=tanh	-			
Dense	▼ Hic	lden,	Output feat	ures=16, Activation=tanh				
Dense	▼ Hic	lden,	Output feat	ures=32, Activation=tanh				
Dense	▼ Hic	lden,	Output feat	tures=8, Activation=tanh				
Dense	▼ Ou	tput,	Output feat	ures=1, Activation=tanh				
* I I	:= \				-			
⊥ ↓ ⊤)utput featu	res:	1						
ctivation.	100.	t	anh	-				
aver configu	uration:	[2,8	3,16,32,8,1]					
Data				 Training and Valida 	ition			
Data				Method:	Adam 🔻			
Data source:		Fil	e	Learning rate:	1e-3			
Decimal sepa	arator:	Po	int	Weight decay:	0			
		USE	RS\Desktor	Batch size:	512			
ilename:		C F	Browse 🔻 Refresh	Loss function:	Root-mean-square error 🔹			
✔ Ignore N	aN/Inf	data p	points	Random seed type:	Fixed •			
/ Data Co	lumn	Setti	ngs	Random seed:	0			
			5	Train on GPU(5)				
Column	Туре		Settings	- Stop condition				
x	Argu	↓ ▼	Name=x1	Number of epochs:	1000			
t	Argu	. ▼	Name=x2	Validation data				
u	Fun	c 🔻	Name=dr	Validation data:	Random sample of data values 🔹			
				Validation data fraction:	0.1			
				Random seed type:	Fixed •			
				Random seed:	0			

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table
 - Data Columns: arguments: x, t function value: u
 - Layers and Activations:[2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation: Default settings + Train on GPU First 10000 epochs with rate 1e-3
 Other 5000 epochs with rate 1e-4 Continue Training
- Plot DNN in the x-t-u space

1	∎↓ ▼	+□‡ ≣• ₹ •	Setting: Global Defin	S nitior	าร	De
20	MSO	L.mph <i>(root)</i>				Lak
or 1 cc	Pi a=	Parameters Variables Equation Contributions	Þ			*
		Functions	•	f(X) Q	Analytic	
u n u	in the second se	Geometry Parts Mesh Parts Default Model Inputs Materials Load and Constraint Group Thermodynamics Parameter Estimation Extra Dimensions Show More Options Node Group Group by Type	> > > > >		Interpolation Piecewise Gaussian Pulse Ramp Rectangle Step Triangle Waveform Normal Distribution Random External MATLAB	↑ Ou Act
	?	Help	F1	Ø	Elevation (DEM)	Da
				 ▲ ※ ※ √ ∧ ♦ 	Image Least-Squares Fit Gaussian Process Polynomial Chaos Exp Deep Neural Network Partial Fraction Fit Function Switch	De File
						х
						t
						u

	Setting Deep Neura Plot 🗟 Label: Deep	S al Netw Create P p Neural	ork Plot	r∞ Train Mo work 1	odel 🛗 Continue Training	
	✓ Layers					
	₩ Туре	Sett	tings			
	Input	▼ Inpu	ıt, İnj	put features	5=2	
	Dense	▼ Hide	den,	Output feat	ures=8, Activation=tanh	
	Dense	▼ Hide	den,	Output feat	ures=16, Activation=tanh	
	Dense	▼ Hide	den,	Output feat	ures=32, Activation=tanh	
	Dense	▼ Hide	den,	Output feat	ures=8, Activation=tanh	
	Dense	▼ Out	put, (Output feat	ures=1, Activation=tanh	
	↑ ↓ + Output featu	ires:	1			
	Activation:		ta	anh	•	
	Layer configu	uration:	[2,8	,16,32,8,1]		ation
	✓ Data					
	Data source:		File	e	Method:	Adam
	Decimal sepa	arator:	Po	int	Learning rate:	1e-3
			USE	RS\Desktor	Weight decay:	0
	Filename:	L	🗁 e	Browse 🔻	Batch size:	512
p			C F	Refresh	Loss function:	Root-mean-square error
k	✓ Ignore N	aN/Inf d	ata p	oints	Random seed type:	Fixed
	 Data Co 	olumn S	etti	ngs	Random seed:	0
		Turne		Cattings	Stop condition	~
l	Column: Type Settings		Number of epochs:	1000 6		
	+	Arqu	•	Name=x2	- Validation data	
	u	Func	•	Name=dr	Validation data:	Random sample of data values
					Validation data fraction:	0.1
					Random seed type:	Fixed
					Random seed:	0

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table
 - Data Columns:
 arguments: x, t
 function value: u
 - Layers and Activations:[2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation: Default settings + Train on GPU First 10000 epochs with rate 1e-3 Other 5000 epochs with rate 1e-4
 Continue Training
- Plot DNN in the x-t-u space

						S
		*□#	Setting	S		D
	∎↓ ▼		Global Defi	nitior	าร	0)
-		C				La
20	MSC	L.mph <i>(root)</i>				
ەr 1	Pi	Parameters				Ť
	a=	Variables				
СС		Equation Contributions	•			
		Functions	•	T(X) Q	Analytic	
		Geometry Parts	•	<u>л</u>	Interpolation	
u		Mesh Parts	•	\wedge	Piecewise	
	<	Default Model Inputs		Л	Gaussian Pulse	
n		Materials	•	/	Ramp	
ŭ		Load and Constraint Group	os 🕨		Rectangle	
		Thermodynamics	•		Step	1
		Parameter Estimation		Wayoform	0	
		Extra Dimensions		Normal Distribution	A	
	θ	Show More Options			La	
	t :	Node Group		External	~	
	籉	Group by Type		Ę,	MATLAB	
	?	Help	F1	9	Elevation (DEM)	D
					Image	D
				1	Least-Squares Fit	
				1.0	Gaussian Process	Fi
			зў.	Polynomial Chaos Exp		
			\sim	Deep Neural Network		
					Partial Fraction Fit	~
				$\hat{\mathbb{N}}$	Function Switch	Þ
						х
						+
						Ľ

	Setting	S			~						
	Deep Neura	Deep Neural Network									
	🔍 Plot 🗔	💷 Plot 🐻 Create Plot 🚾 Train Model 🛗 Continue Training									
	Label: Dee	p Neur	al Net	work 1							
	✓ Layers										
	₩ Туре	Se	ttings			1					
	Input	▼ Inp	ut, In	put features	5=2						
	Dense	▼ Hic	lden,	Output feat	ures=8, Activation=tanh						
	Dense	▼ Hic	lden,	Output feat	ures=16, Activation=tanh						
	Dense	▼ Hic	lden,	Output feat	ures=32, Activation=tanh						
	Dense	▼ Hic	lden,	Output feat	ures=8, Activation=tanh						
	Dense	▼ Ou	tput,	Output feat	ures=1, Activation=tanh						
		:= \									
	↓↓ +		1								
	Activation:	nes.	+	anh	-						
	Laver configu	uration	12.8	16 32 8 1]							
	Layer connige		[_,<	, 10,02,0,1]	 Training and Valida 	ation					
	✓ Data				Mathad:	Adam	•				
	Data source:		File Point USERS\Desktor		Learning rate:	1e-3 7					
	Decimal sepa	arator:			Weight decay:	0	۲				
					Batch size:	512	۲				
	Filename: 🔽 Browse 🔻				Loss function:	Root-mean-square error	•				
0	✓ Ignore N	aN/Inf	data p	points	Random seed type:	Fixed	•				
K	 Data Co 	olumn	Setti	nas	Random seed:	0					
	**			5	✓ Train on GPU						
i	Column	Туре	Type Setti		- Stop condition						
	Х	Arg	u 🔻	Name=x1	Number of epochs:	1000					
	t	Arg	u 🔻	Name=x2	Validation data						
	u	Fun	с 🔻	Name=dr	Validation data:	Random sample of data values	•				
					Validation data fraction:	0.1					
					Random seed type:	Fixed	•				
					Random seed:	0					

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table
 - Data Columns:
 arguments: x, t
 function value: u
 - Layers and Activations:[2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation: Default settings + Train on GPU First 10000 epochs with rate 1e-3 Other 5000 epochs with rate 1e-4 Continue Training (8)
- Plot DNN in the x-t-u space

	∎↓ ▼	C Settin	n gs Definitio	ons	Settin Deep Neu Plot R Label: De	
:0	MSO	L.mph <i>(root)</i>				
۶r 1	Pi	Parameters			✓ Layers	
	a=	Variables			Туре	
СС		Equation Contributions	•		Input	
		Functions		Analytic	Dense	
		Geometry Parts	• •	Interpolation	Dense	
u		Mesh Parts	•	Piecewise	Dense	
-	≪*	Default Model Inputs		 Gaussian Pulse 	Dense	
n		Materials	• /	Ramp	Donso	
u		Load and Constraint Groups	•	Rectangle	Dense	
		Thermodynamics	•	Step	↑↓+	
		Parameter Estimation	•	Triangle	Output fea	
		Extra Dimensions	•	Waveform	Activation:	
	θİ	Show More Options		Normal Distribution	Laver confi	
		Node Group	/N/	Random		
	=	Group by Type	Lt _e	External	✓ Data	
			\u03e4	MATLAB	Data sourc	
	2	Help F	1 (@	Elevation (DEM)	Decimal se	
				Image		
			17	Least-Squares Fit	Filename:	
			20	Gaussian Process		
			Ŀ.	Polynomial Chaos Exp	✓ Ignore	
			•	 Deep Neural Network 		
				 Partial Fraction Fit 		
			ŔŶ	Function Switch	* Colum	
					х	
					t	
					u	

ngs								
eura	al Netwo	ork						
5	🗟 Create Plot 🔯 Train Model 🔯 Continue Training 🖉							
eep	eep Neural Network 1							
rs								
	Sett	ings	/					
	 Inpu 	t, Inj	put features	5=2				
	▼ Hide	len,	Output feat	ures=8, Activation=tanh	1			
_	▼ Hidd	len,	Output feat	ures=16, Activation=tanh	1			
	▼ Hidd	len,	Output feat	ures=32, Activation=tanh	1			
	▼ Hidd	len,	Output feat	ures=8, Activation=tanh	1			
	 Outp 	out, (Output feat	ures=1, Activation=tanh	1			
_					1			
+								
atu	res:	1						
1		ta	anh	•				
figu	iration:	[2,8	,16,32,8,1]	Turinin a surd Valida	dia a			
				 Training and valida 	ition			
co.		Fil	•	Method:	Adam 🔻			
	uratori [De	int	Learning rate:	1e-3			
epa				Weight decay:	0			
	L			Batch size:	512			
		C F	Refresh	Loss function:	Root-mean-square error 🔹			
e Na	aN/Inf da	ata p	ooints	Random seed type:	Fixed •			
Со	lumn S	ettii	ngs	Random seed:	0			
g-				✓ Train on GPU				
nn:	Type Settings		Settings	Stop condition				
	Argu	•	Name=x1	Number of epochs:	1000			
	Argu ▼ Name=x2		Name=x2	Validation data				
Func ▼ Name=dr		Name=dr	Validation data:	Random sample of data values 🔻				
				Validation data fraction:	0.1			
				Random seed type:	Fixed •			
				Random seed:	0			

NCOMSOL

Baking a Deep Neural Network (DNN)

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table
 - Data Columns: arguments: x, t function value: u
 - Layers and Activations: [2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation: Default settings + Train on GPU First 10000 epochs with rate 1e-3 Other 5000 epochs with rate 1e-4 Continue Training
- Plot DNN in the x-t-u space (9

					Sotting				
IIII ▼ DMSC Pi	 □ ↓ ↓ ↓ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓ □ ↓<td>Settings Global Defin</td><td>S nitior</td><td>ns</td><td>Deep Neura Plot a Label: Dee Layers</td><td>Create p</td><td>) Plot Il Net</td><td>[∱]∰ Trair twork 1</td>	Settings Global Defin	S nitior	ns	Deep Neura Plot a Label: Dee Layers	Create p) Plot Il Net	[∱] ∰ Trair twork 1	
a=	Variables				► Туре	Set	tings	5	
	Equation Contributions	•	f(v)		Input	▼ Inp	ut, In	put feat	
	Functions	•	ă	Analytic	Dense	▼ Hid	den,	Output	
	Geometry Parts	•	Л.	Interpolation	Dense	▼ Hid	den,	Output	
	Mesh Parts	•	~	Piecewise	Dense	▼ Hid	den,	Output	
*	Default Model Inputs		7	Pamp	Dense	▼ Hid	den,	Output	
i	Materials			Rectangle	Dense	▼ Out	put,	Output	
	Load and Constraint Group	os 🕨		Step					
	Thermodynamics			Triangle	↑↓+≣▶				
				Waveform	Output featu	1			
				Normal Distribution	Activation:			anh	
θ				Random	Layer config	uration:	[2,8	8,16,32,8	
t.	Node Group		Fr Fr	External	✓ Data				
臺	Group by Type			MATLAB			F 'I		
?	Help	F1		Elevation (DEM)	Data source:		File		
				lmage	Decimal separator:			Point	
			i. K	Least-Squares Fit			USE	RS\Desk	
			2.	Gaussian Process	Filename:			Browse	
				🔅 🛛 Polynomial Chaos Exp	Refres			Refresh points	
			\sim	Deep Neural Network	g				
			$^{\sim}$	Partial Fraction Fit	 Data Column Set 		Setti	ngs	
			Ŕ	Function Switch	Column:	Туре		Settin	
					X	Argu	•	Name	
					t	Argu	•	Name	
					u	Fund	•	Name	

* •							
n Mo	n Model 🚾 Continue Training						
ures	s=2						
feat	ures=8, Activation=tanh						
features=16, Activation=tanh							
feat	ures=32, Activation=tanh						
feat	ures=8, Activation=tanh						
feat	ures=1, Activation=tanh	1					
11	• • • • • • • • • • • • • • • • • • •						
→ Training and Validation							
	Method:	Adam					
	Learning rate:	1e-3					
	Weight decay:	0					
ctor -	Batch size:	512					
·	Loss function:	Root-mean-square error					
	Random seed type:	Fixed •					
	Random seed:	0					
	✓ Train on GPU						
gs	- Stop condition						
=x1	Number of epochs:	1000					
=x2	Validation data						
=dr	Validation data:	Random sample of data values					
	Validation data fraction:	0.1					
	Random seed type:	Fixed •					
	Random seed:	0					

Baking a Deep Neural Network (DNN)

- Export Data
- Functions: Deep Neural Network
 - Data: exported *.txt table
 - Data Columns: arguments: x, t function value: *u*
 - Layers and Activations: [2, 8, 16, 32, 8, 1] with *tanh* activation
 - Training and Validation: Default settings + Train on GPU First 10000 epochs with rate 1e-3 Other 5000 epochs with rate 1e-4 Continue Training
- Plot DNN in the x-t-u space





Line Graph: Dependent variable u Line Graph: Deep Neural Network



Line Graph: Dependent variable u Line Graph: Deep Neural Network



More Realistic Example



<u>kozisek@humusoft.cz</u> www.linkedin.com/in/martinkozisek/

More Realistic Example: Less Training Data

- Design Space x-t-u is perfect showcase, but you would probably use interpolation instead of DNN.
- DNN is technology for larger sets of data and more complex systems.
- Imagine x-y-z-T-P training data space:
 - Air Cooled BESS
 - Training data for 55 parameter values (105 M rows in the table)
 - DNN [5,50,100,200,300,200,100,50,1]
 - Digital Twin in application
- In this case you will have probably sparse space of training data



More Realistic Example: Less Training Data

- 1D Component
 - Interval
- ID General Form PDE
 - Filling the equation into the template
 - Initial Condition Settings
 - Dirichlet Boundary Condition
- User Controled Mesh
 - Edge with Distribution
 - 1000 elements 50 elements
- Time Dependent Study
 - Output times: range(0,1/50,1)
 - Adaptive mesh refinement



More Realistic Example: Less Training Data

- 1D Component
 - Interval
- ID General Form PDE
 - Filling the equation into the template
 - Initial Condition Settings
 - Dirichlet Boundary Condition
- User Controled Mesh
 - Edge with Distribution
 - 1000 elements 50 elements
- Time Dependent Study
 - Output times: range(0,1/50,1)
 - Adaptive mesh refinement

What if there is only a small amount of data due to high computational cost? Training data are either missing or have bad precision in potentially important regions! DNN has no chance to approximate the real-physics correctly.



More Realistic Example: Less Training Data

- 1D Component
 - Interval
- ID General Form PDE
 - Filling the equation into the template
 - Initial Condition Settings
 - Dirichlet Boundary Condition
- User Controled Mesh
 - Edge with Distribution
 - 1000 elements 50 elements
- Time Dependent Study
 - Output times: range(0,1/50,1)
 - Adaptive mesh refinement

What if there is only a small amount of data due to high computational cost? Training data are either missing or have bad precision in potentially important regions! DNN has no chance to approximate the real-physics correctly.

