Using AI to predict excavation behaviour without calculations

Sacha Wattel, GeoMod
Prof. Stéphane Commend, iTEC, HEIA-FR, HES-SO
Prof. Jean Hennebert, iCoSys, HEIA-FR, HES-SO
Prof. Pierre Kuonen, iCoSys, HEIA-FR, HES-SO
Prof. Laurent Vulliet, LMS, EPFL

Purpose of this work

- Have an almost instantaneous estimation of excavation behaviour
- f(excavation geometry, soils characteristics, support system) -> excavation behaviour

- Why ?
  - Fast and creative pre-design
  - Automatic pre-design

- The goal is not to replace Zsoil or ZSWalls

First step

- Proof of concept :
  - one soil
  - freestanding sheet pile wall
  - excavation depth between 2m and 8m
  - No water

- f(exc, L, E, phi, c, A, I) -> (convergence, maximum settlement, maximum wall deviation, maximum bending moment)

Processus

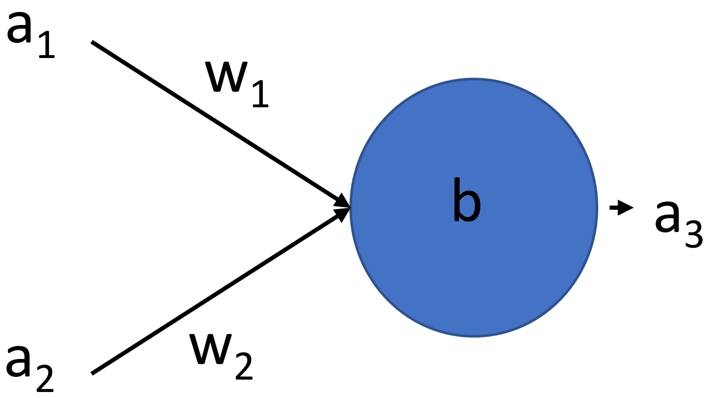
- Supervised learning : learning a function that matches input to output (function fitting)
- Inputs and outputs obtained from a synthetic databases of ZSWalls simulations
- Neural network is the function to be fitted

Database

- Automatic generation of the inpw files for ZSWalls
- Batch command to execute the computation
- Results extracted with zstools developed by M. Preisig

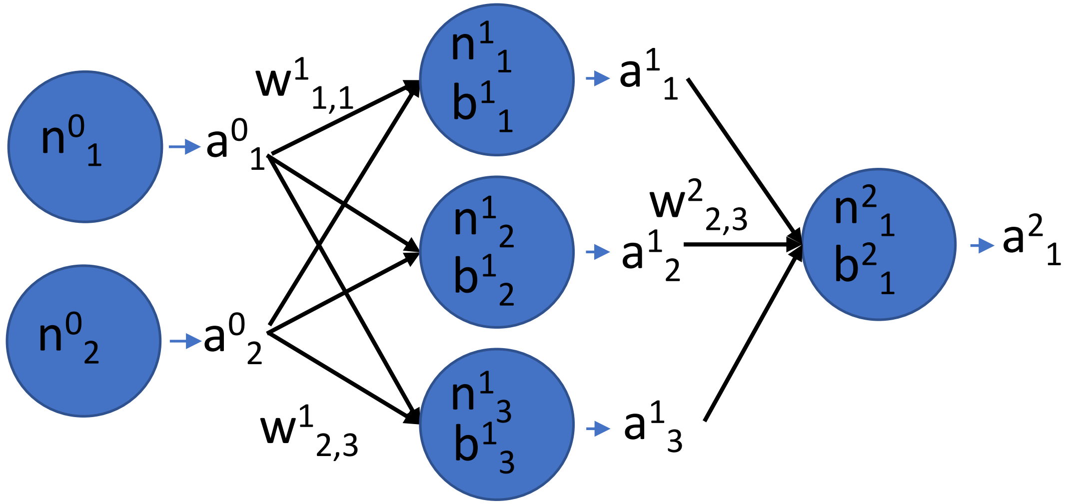
Neural networks

- A neuron



- $a_3 = f(w_1*a_1+w_2*a_2+b)$
- f can be :
  - Hyperbolic tangent
  - Sigmoid
  - ReLU

- A neural network



- The parameters to be learned are the weights w and bias b
- $a_i^l = f\left(\sum_k w_{k,i}^l a_k^{l-1} + b_i^l\right)$

How does a neural network learn ?

Random parameters

- sampled from a carefully chosen distribution to help training

Forward pass

- with some samples from the database

Computation of the error

- Mean squared error
- Mean absolute error
- Binary cross entropy

Backpropagation

- Chain rule to compute the gradient of the error w.r.t the parameters

Gradient Descent

- on the parameters

Why Neural networks ?

- Universal function estimator :
  - Given enough neurons, a neural network can represent any continuous function at any precision wanted
- More neurons -> more «representational power» : can learn more complex function

- Good (but not well understood) generalisation capacity between training points
- But risk of overfitting ! (learning by-rote or learning the noise)

Architecture of neural networks

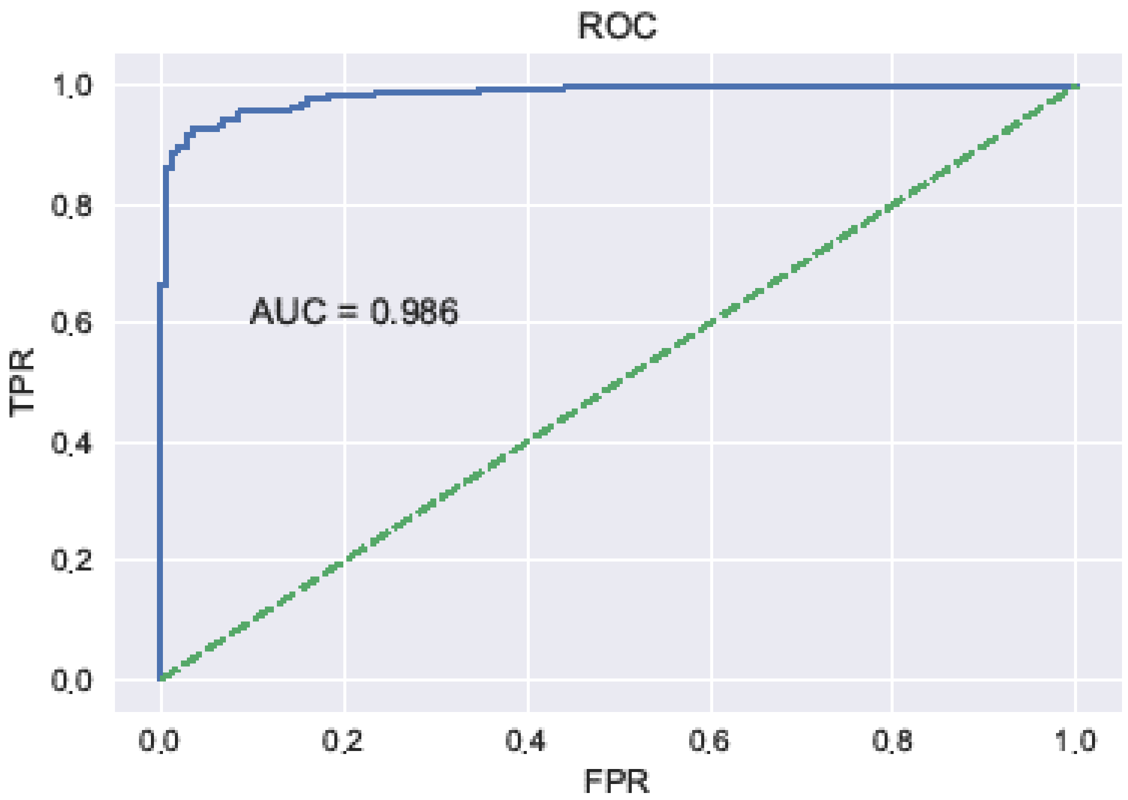
- To predict convergence : one hidden layer of 10 neurons
  - Output a number between 0 and 1, the closer to 1, the likelier the excavation holds

- To predict displacements and moments :
  - One network per output
  - Three architectures tested :

Name	Number of hidden layers	Number of neurons per hidden layer
10	1	10
30	1	30
30red	3	30-20-10

Results : Convergence

Cut-off point	Accuracy	PPV	NPV
-	%	%	%
0.5	92.9	94.52	90.26
0.9	88.7	98.67	77.73
0.95	99.2	99.18	72.43
0.99	79	100	64.2



PPV : Positive Predicted Value : fraction of positive predicted values that are correct
NPV : Negative Predicted Value : fraction of negative predicted value that are correct

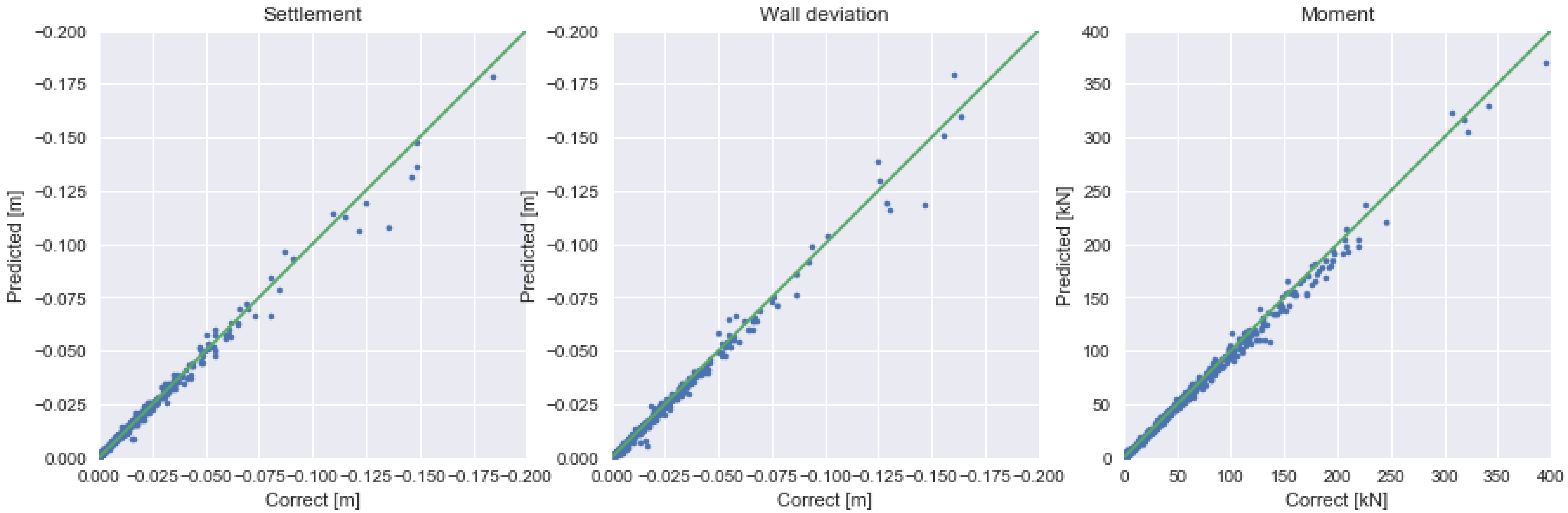
TPR : True Positive Rate : fraction of positives classified correctly
FPR : False Positive Rate : fraction of negative classified incorrectly

Results : Displacements and moment

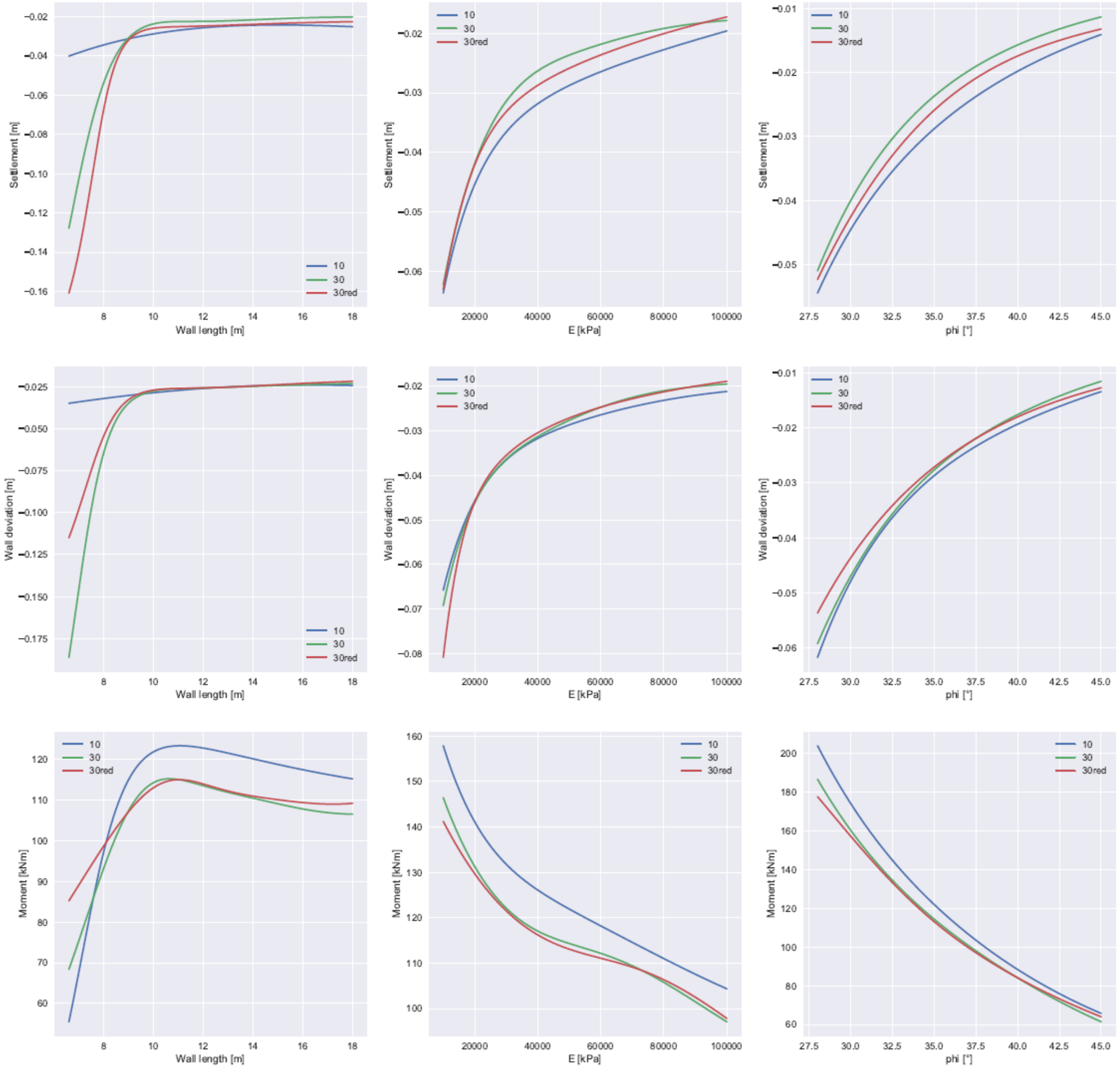
MAE : Mean Absolute Error **MRE** : Mean Relative Error **RE95** : 95 percentile of the Relative Error
_tr : truncated : absolute errors under 2 mm or 5 kN omitted

	Model	R^2	MAE [m]	MRE	RE95	MRE_tr	RE95_tr
Settlement	10	0.983	1.7E-03	1.52	8.21	0.08	0.21
	30	0.988	1.3E-03	1.07	5.90	0.06	0.15
	30-20-10	0.989	1.4E-03	1.40	7.00	**0.05**	**0.12**
Wall deviation	**Model**	**R^2**	**MAE [m]**	**MRE**	**RE95**	**MRE_tr**	**RE95_tr**
	10	0.982	1.3E-03	0.22	0.88	0.04	0.17
	30	0.991	1.1E-03	0.16	0.54	0.02	0.14
	30-20-10	0.991	9.8E-04	0.12	0.37	**0.02**	**0.11**
Moment	**Model**	**R^2**	**MAE [kN]**	**MRE**	**RE95**	**MRE_tr**	**RE95_tr**
	10	0.995	2.7	0.13	0.34	0.02	0.10
	30	0.995	2.6	0.13	0.30	0.01	0.08
	30-20-10	0.994	2.7	0.12	0.30	**0.01**	**0.08**

Results : Displacements and moment



Results


Settlement [m]
Wall deviation [m]
Moment [kNm]
Wall length [m]
E [kPa]
phi [°]
10
30
30red

Application : Automatic design

A script tries to find the optimal freestanding sheet pile (length and model) for a given excavation

- How :
  - For every sheet pile type from a given catalogue, find the shortest length that satisfies the constraints.
  - Chose the one that minimises the objective function
- Constraints :
  - Settlements and wall deviation inferior to 1/300 * excavation depth
- Objective function (naive):
  - Minimum steel volume

- Results out of 256 cases :
  - 155 cases where a freestanding sheet pile wall could holds
  - Of which :
    - 133 (86%) respected the constraints
    - 11 (7%) slightly violated the constraints
    - 6 (4%) violated the constraints more (max : 21% too much displacements)
    - 5 (3%) did not converge
  - In average, the constraints were respected with a margin of 41%
  - 98 s total computation time for all cases

Conclusion

- Concept proved !
  - Mean Relative Error between ZSWalls and Neural network below 5% and below 12% for 95 predictions out of 100
  - Very fast prediction (~1 ms)

- What's next ? (Current work)
  - Extends the model to multiple soils, different support system (diaphragm wall, struts, anchors)
  - Build a less naive automatic pre-design procedure