Artificial Neural Networks:

Practices, Needs and Future Developments

Ian Flood

University of Florida

Topics Covered:

- Introduction
  - Quick ANN Tutorial
  - Brief History
  - System complexity achieved to date?
- System Fundamentals:
  - Main features of an ANN System
  - Graphical Interpretation of ANN Operation
- Challenges with Current Applied ANNs:
  - Geometric Explosion in Data Set Size
  - Extensibility
  - Black Boxes and Explainability
- Deep Learning:
  - Characteristics
  - What is the Excitement?
  - Example: Convolutional Deep Learning ANNs
  - Transfer and Multi-Task Learning
- Next Generation ANNs:
  - Further Inspiration from Biological Systems
  - Richly Structured Networks and Learning Schemes
  - Example Using Growth Algorithms
- Appendix: Development Methodology:
  1. Strategizing
  2. Collation and Evaluation of Data
  3. Model Development
  4. Model Evaluation and Final Selection
  5. Final Validation
  6. Implementation and Review
- Brief Bibliography:

Introduction

- Desktop computing has had a profound influence on our ability to solve problems. Consider from engineering:
  - Finite Element Method
  - Dynamic Simulation
  - 3D & 4D Visualization
- Yet, the world is full of problems that have defied solution using conventional computing techniques...
- ...problems that can often be solved by people with appropriate training, eg:
  - Legal compliance of engineering designs
  - Identifying fabrication issues from designs
  - Uncoupling superimposed signals

- This class of problems has been a target of artificial intelligence (AI). Two main approaches:

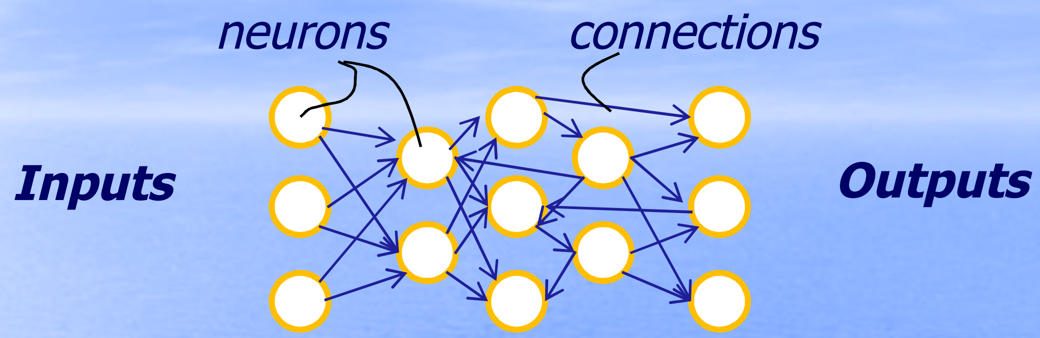
(1) Classical AI (symbol manipulation):

- Attempts to capture essence of human cognition at a **high level**
- Some successes, but **poor learning** capabilities

(2) ANNs (connectionist systems):

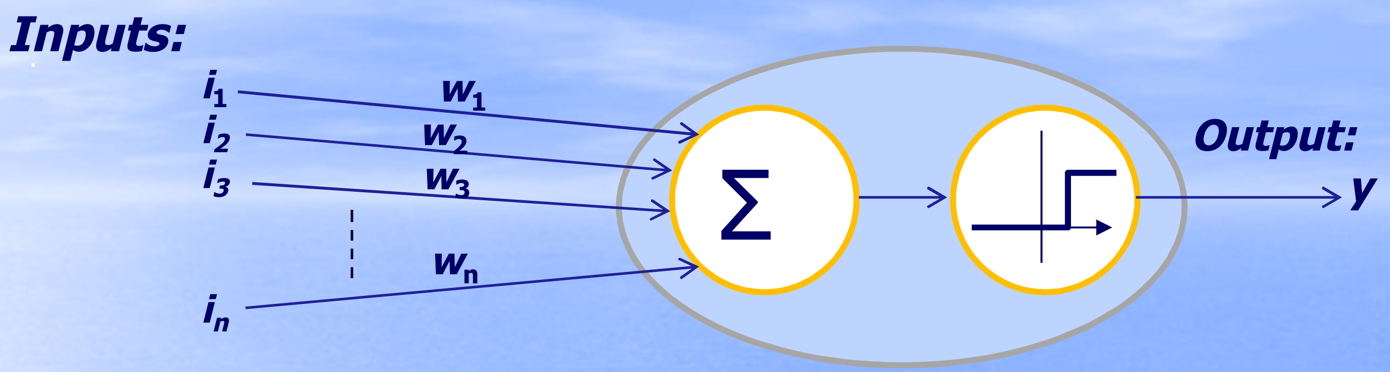
- Emulate operation of the brain from a relatively **low-level** (the neuron)
- Intent is to achieve higher-level human cognition as an **emergent property**
- Some successes, but **failed to move far** beyond low-level problems
  - somewhere just beyond the capabilities of non-linear regression, or pattern recognition/classification
- Yet biological neural systems promise so much more than this.

- Quick ANN tutorial:
  - biologically inspired computing devices composed of many (handful to billions) of neurons connected within a network:



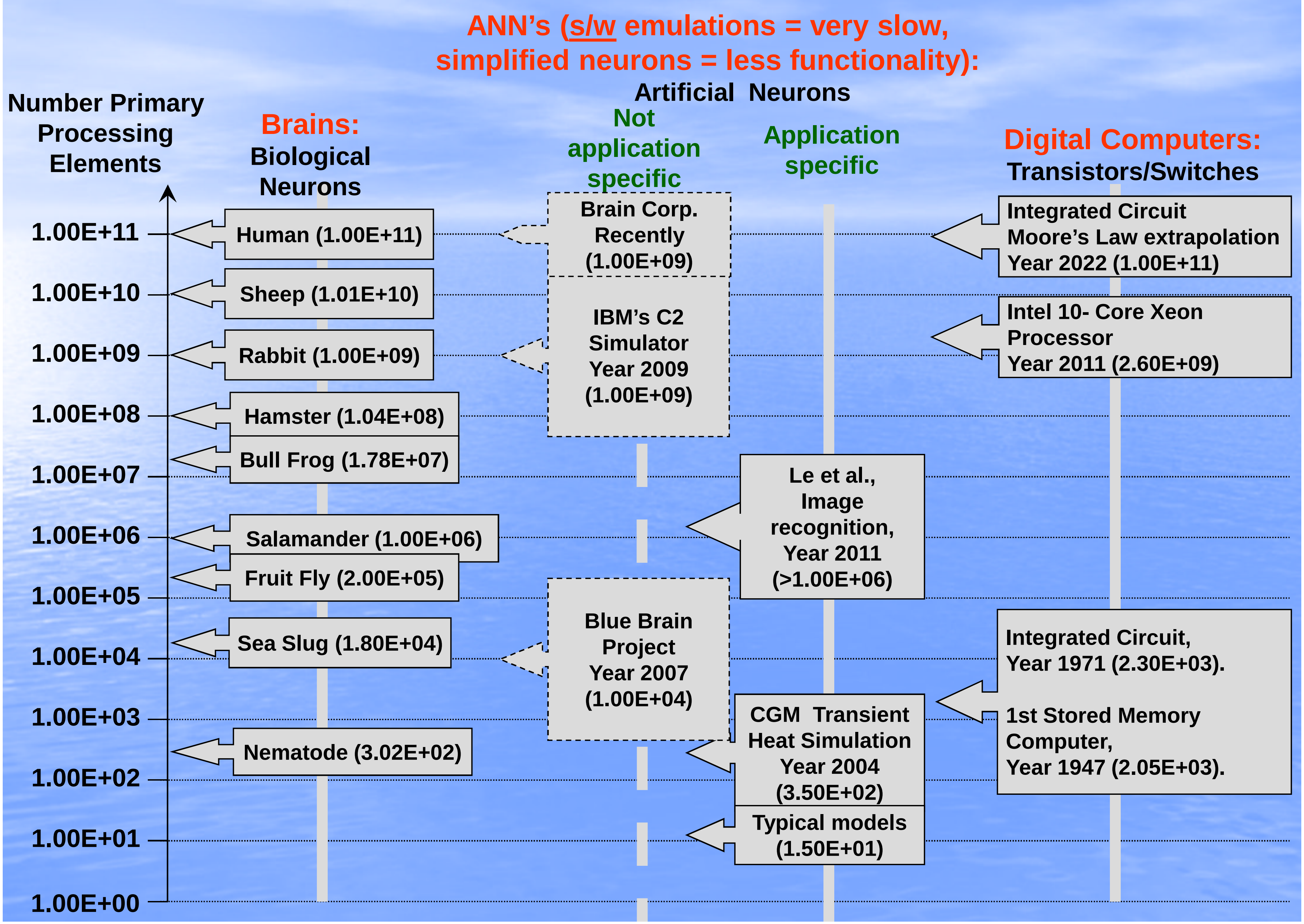
  - each neuron and its connections implement a simple non-linear function, comparable to that of a non-linear regression model
  - all elements of the ANN work together to solve a higher-order problem
  - the broader problem solved by the ANN depends on, for example:
    - the connectivity of the ANN (eg: feedforward),
    - the functions implemented at the neurons and connections (eg: sigmoid & weights)
    - the values of ANN parameters (connection weights, neuron biases, etc...)
  - these parameters are developed through training, often to solve a set of example problems with known solutions (supervised training).

- Brief History:
  - **1943**: McCulloch-Pitts model of neuron (binary 0/1 weights & output)



  - **Late 1940s**: Hebbian learning (correlates weight change with activity)
  - **1957**: Rosenblatt, the Perceptron (real weights, learning rule)
  - **1969**: Minsky & Papert publish Perceptrons (show cannot solve the XOR problem and computationally too expensive = 1st ANN winter)
  - **1986:** Rumelhart et al. rediscover and popularize Backpropagation (possible to train multi layered non-linear networks)
  - **2000's**: Generic ANN tools hit a glass ceiling applications-wise.. Other AI techiques often outperformed. 2nd ANN winter.
  - **Deep Learning:** multi layered. Conceptually been around since the beginning, but 2010's started to outperform other AI approaches (GPUs, training techniques, and specific architectures).

Compare complexity: ANNs vs. Biological Systems vs. General Purpose Digital Computer

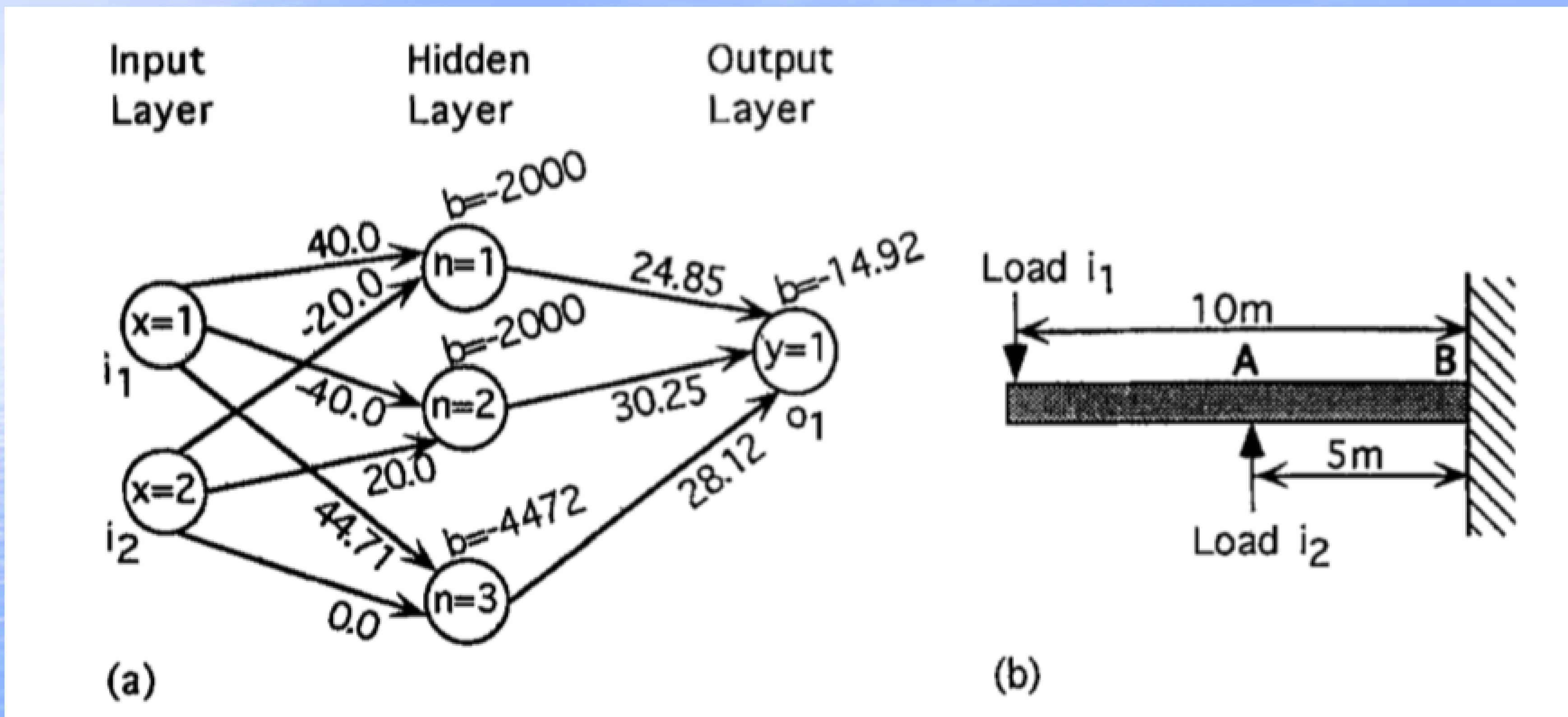


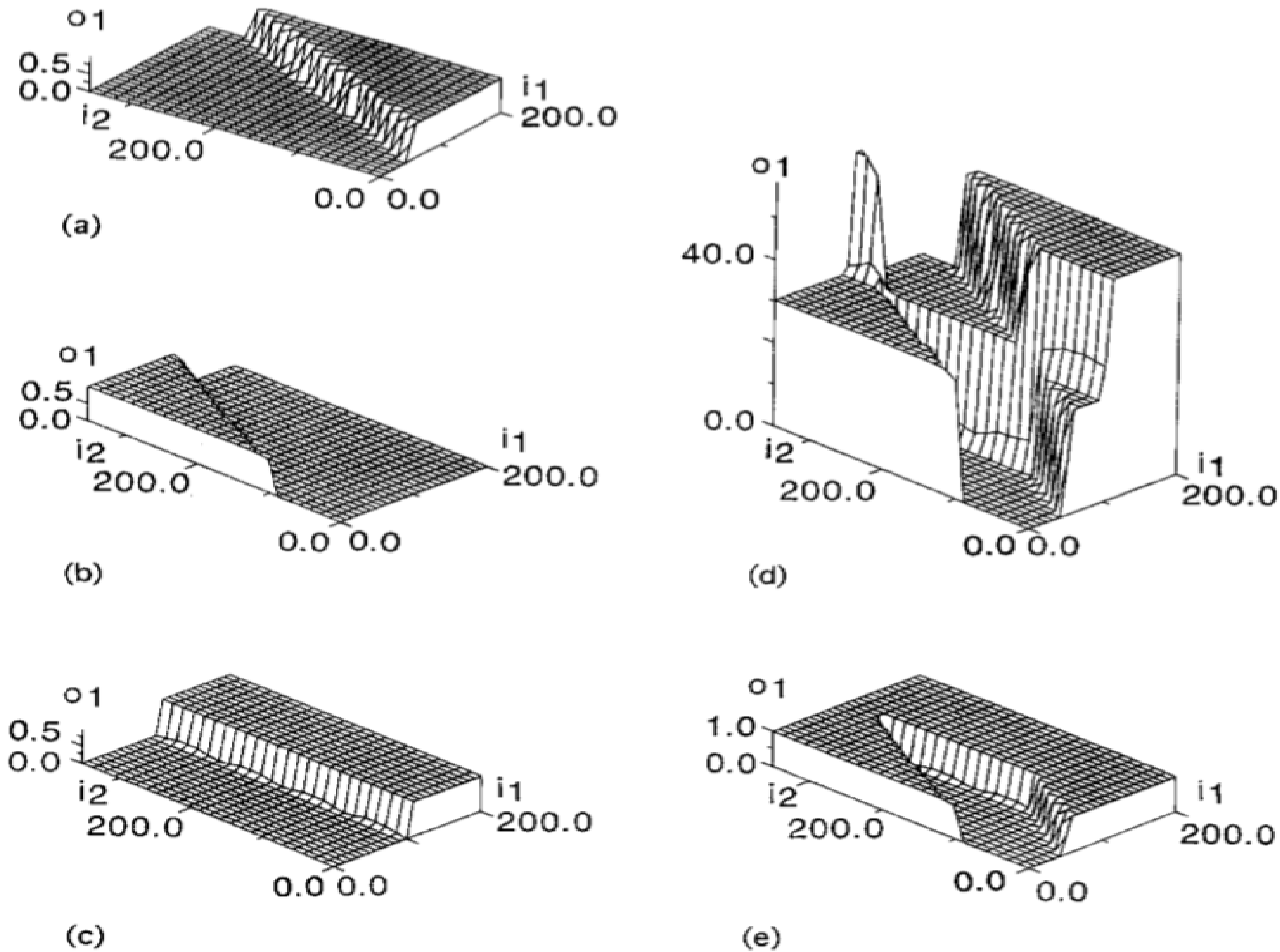
- Of course, the number of primary processing units usefully employed is an overly simplistic measure of complexity:
  - artificial neurons >> complicated than transistors,
  - & biological neurons >> complicated than artificial neurons.
- However, this comparison tells us:
  - ANNs may have reached complexity of the Salamander (but remember these are simplified neurons and simulated therefore slow);
  - the biological model indicates ANN’s have a great potential yet to be realized.
- It is possible, today, to build ANNs with billions of neurons:
  - however, we don’t know how to make these massive networks perform useful tasks.
  - we know how to use greater network size to achieve greater precision, but not to achieve greater functionality.

System Fundamentals

- Main Features of an ANN System:
  - Data structures (input and output):
    - values (real, binary, enumerative)
    - format (order and interpretation – absolute or relative)
  - Connectivity:
    - feedforward fully connected
    - recursive (feedback)
    - number of layers…
  - Mode of operation:
    - synchronous/asynchronous firing
    - value or pulse rate output
    - type of activation function, type of weights
  - Method of Training:
    - supervised (with example input to output mappings)
    - supervised (with example inputs and post evaluation of performance)
    - unsupervised
    - staged or all-at-once learning…

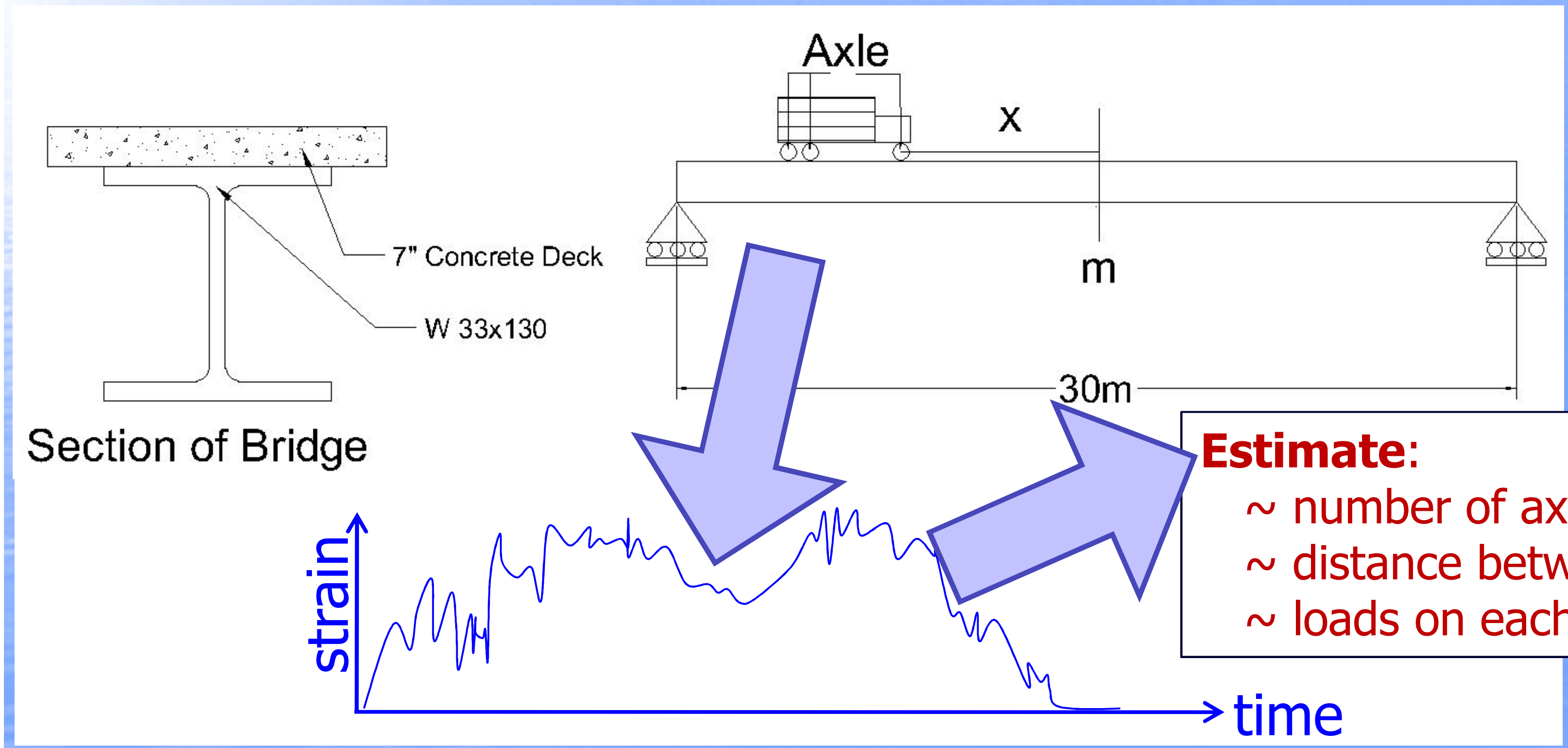
- Graphical understanding example: Does the maximum Bending Moment induced in the cantilever by loads i1 and i2 exceed 500 kNm?




o1
0.5
0.0
i2
200.0
0.0 0.0
i1
200.0
(a)
o1
0.5
0.0
i2
200.0
0.0 0.0
i1
200.0
(b)
o1
0.5
0.0
i2
200.0
0.0 0.0
i1
200.0
(c)
o1
40.0
0.0
i2
200.0
0.0 0.0
i1
200.0
(d)
o1
1.0
0.0
i2
200.0
0.0
0.0
i1
200.0
(e)

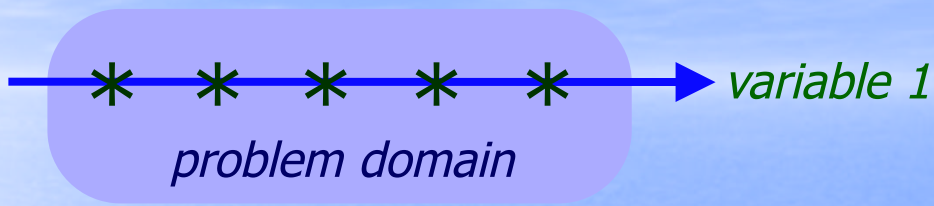
Challenges with Current Applied ANNs

- Truck weigh-in-motion (WIM) is a good benchmark problem for ANNs (encompasses the main challenges)
  - Estimating truck axle loads and spacings from the stress or strain envelopes they induce on bridge members (WIM):

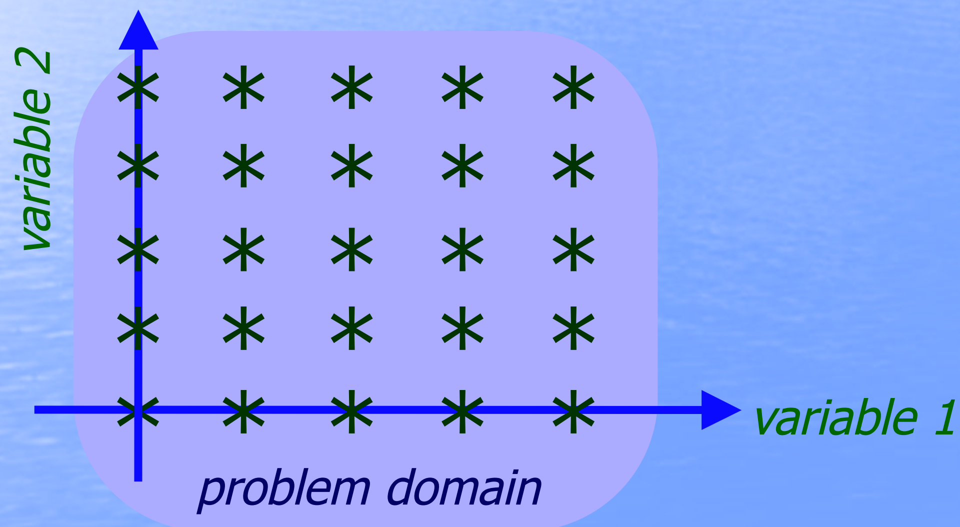


Estimate:
~ number of axles,
~ distance between axles
~ loads on each axle.

- **Challenge 1**: Geometric increase in required number of training examples with linear increase in number of independent variables:
  - say we need a density of 5 training examples across the range of an independent variable:



  - with two independent variables this increases to 5^2=25 examples;

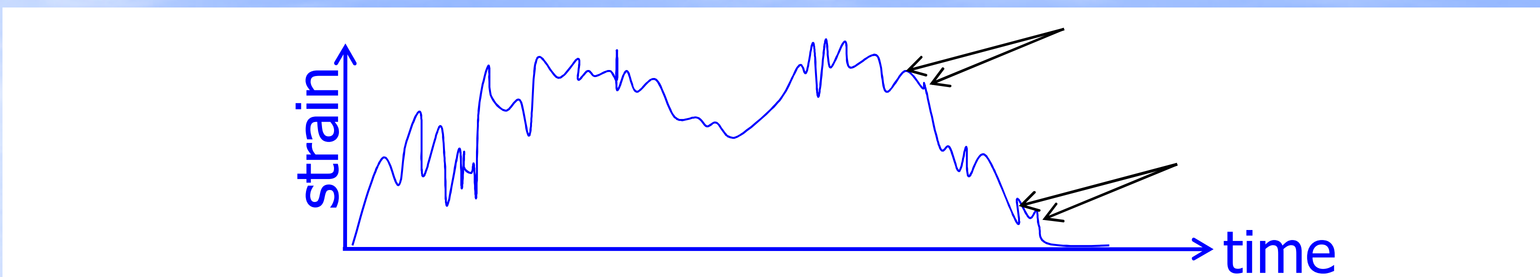


  - the limit is usually 5 or 6 independent variables: 5^6=15,625 examples

# independent variables:	1	2	3	4	5	6	7	8	9
# observations (5/variable):	5	25	125	625	3,125	15,625	78,125	390,625	1,953,125

– For independent variables that are partially/fully correlated, the increase in training examples will only be linear

– For WIM, strain readings made close in time are strongly correlated.



– an ANN implementation had ~100 strain inputs, and only needed a few thousand training examples (not: 5^{100}).

– However, the implementation only worked for ONE bridge

– Considering a range of bridges would have required the introduction of many uncorrelated independent variables, describing:

- Geometric parameters (length, width, skew)
- Number of lanes, supports,
- Materials used in construction, etc

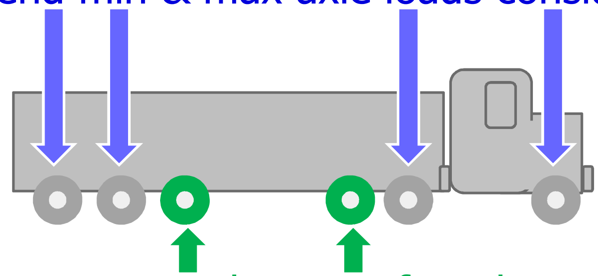
- **Challenge 2**: Extensibility of the ANN solution (easy direct or indirect extension of the ANN to new variants of the problem).

  Broadly this may involve an ability to:

  - Interpolate, extrapolate, and re-calibrate the **values** at the inputs.
  - Change the structure and format of the **variables** at the inputs.

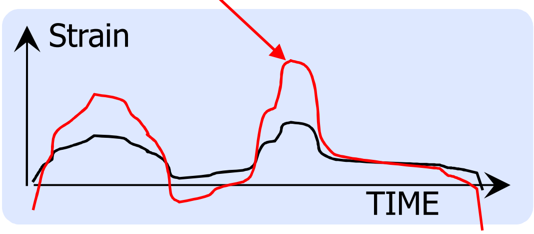
- E.g. increase the scope of application of an ANN:
  - ANNs are developed to solve a class of problems
  - …often there is a need to **extend** the class of problems solved (increase the functionality of the model)

extend min & max axle loads considered (extend values of dependent variables)



extend range of truck types considered
(extend model internal structure, extend number of dependent variables)

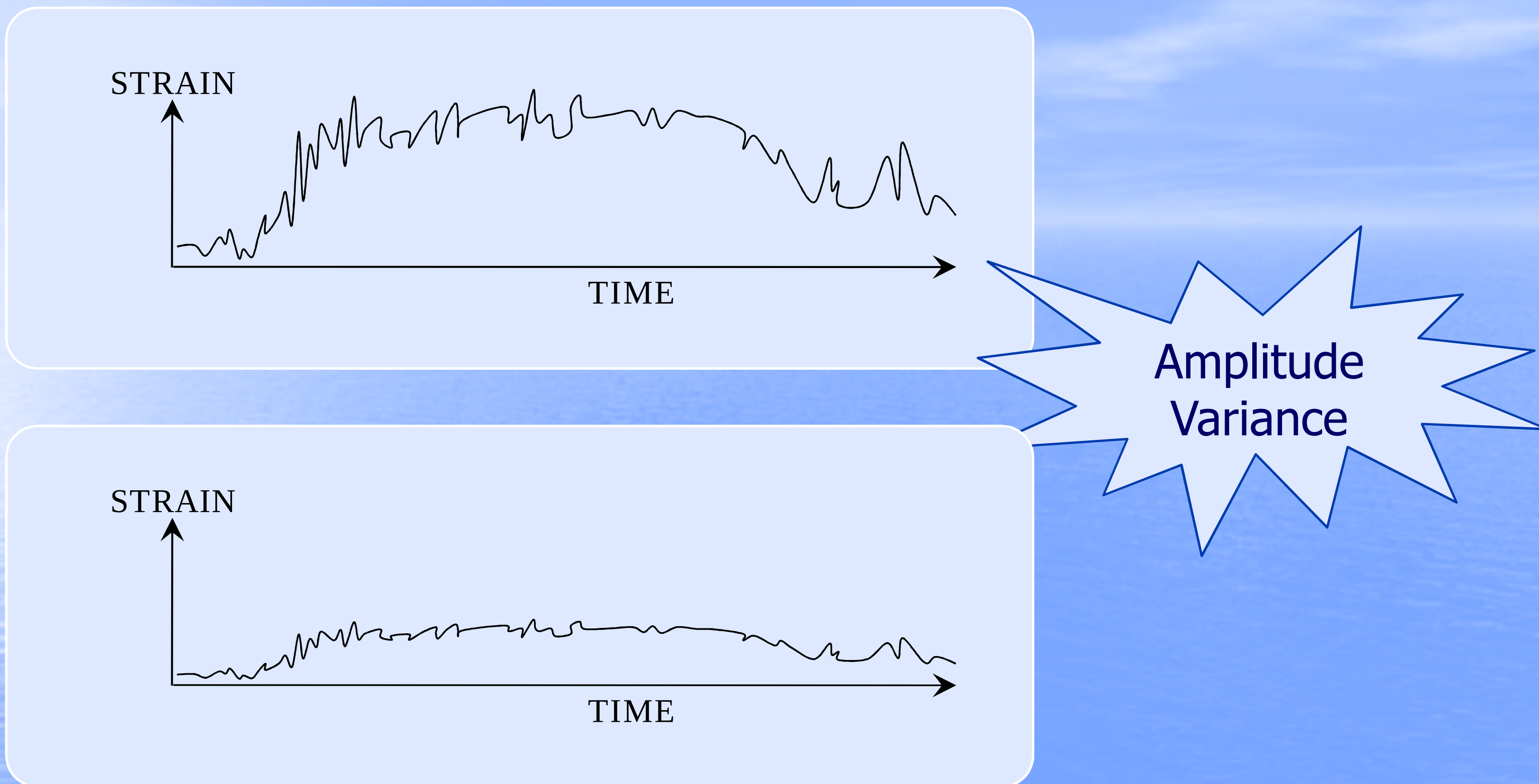
extend range of values for strain readings considered (extend values of independent variables)



Others:
extend bridge lengths considered,
extend number of lanes, etc…

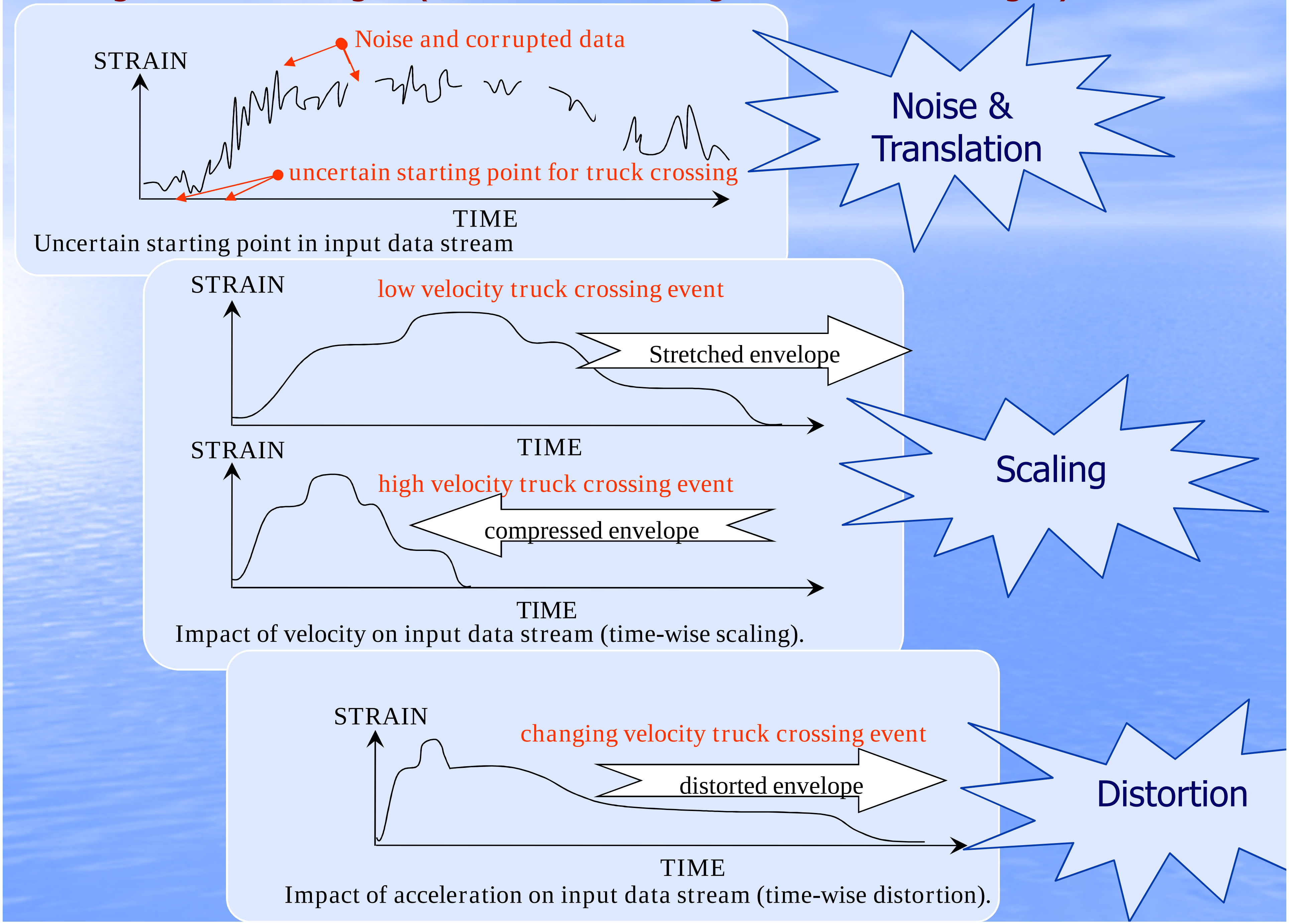
– extension should be achievable without the model-user having to completely rebuild the existing model

– E.g. amplitude variance (value changes):



– ambiguous: could be due to lighter loads or travelling in adjacent lane
– therefore, need to sample strain at multiple locations on bridge
– Brain has no problem with these issues.

- E.g. Format changes (order and meaning of variables changes):

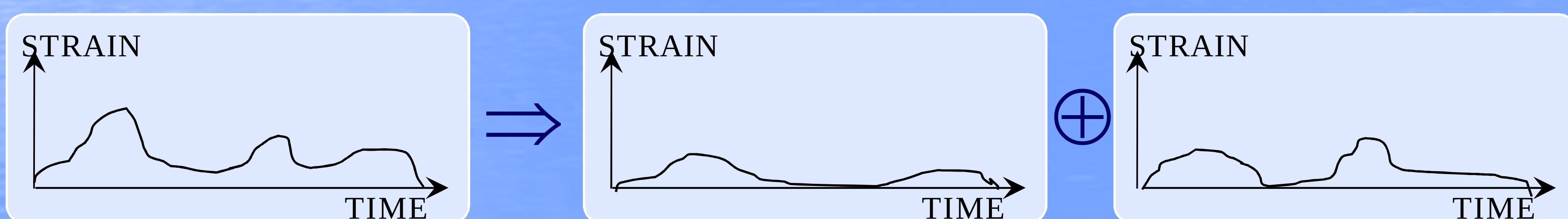


– E.g. superposition of signals and noise:

people are very good at following one conversation in a room full of many concurrent conversations.

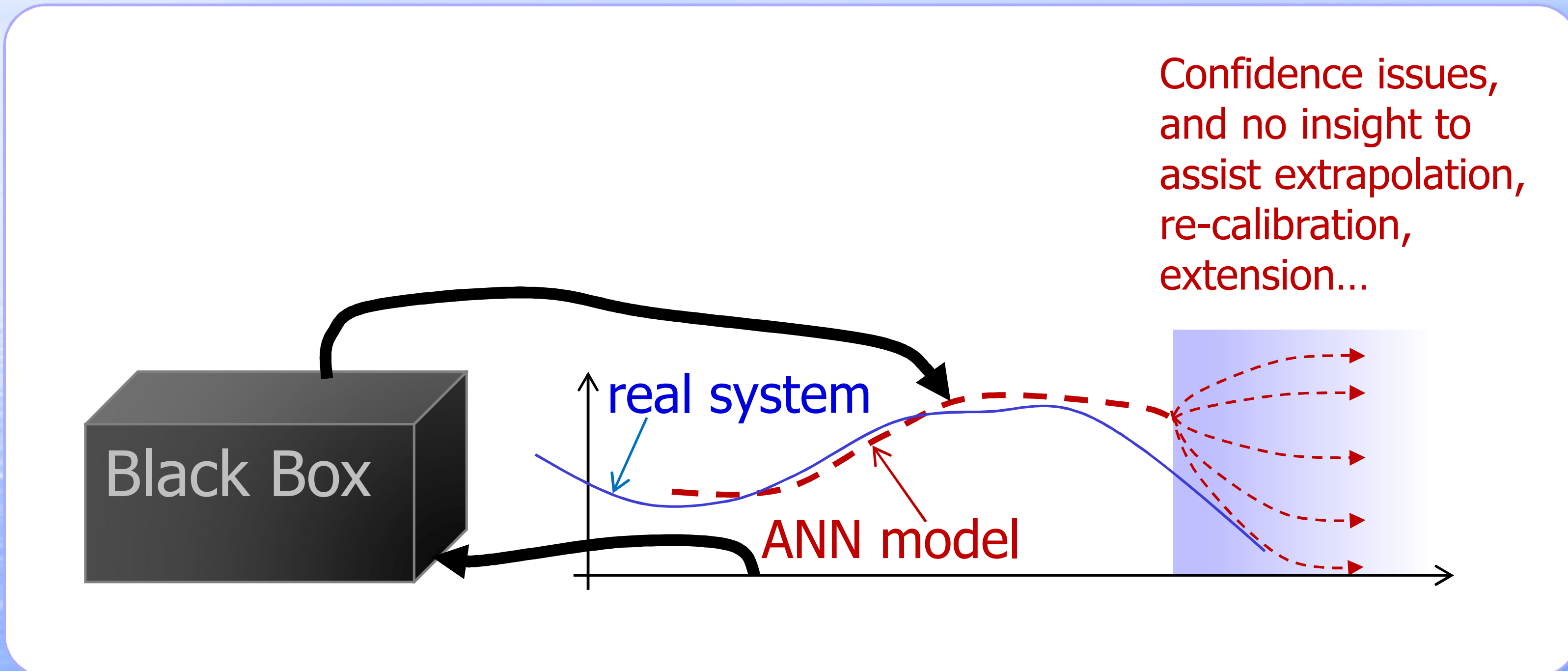
for truck WIM this is somewhat analogous to uncoupling the strain envelopes created by concurrent truck crossings:

- travelling in parallel lanes (same or opposite directions)
- travelling in the same lane



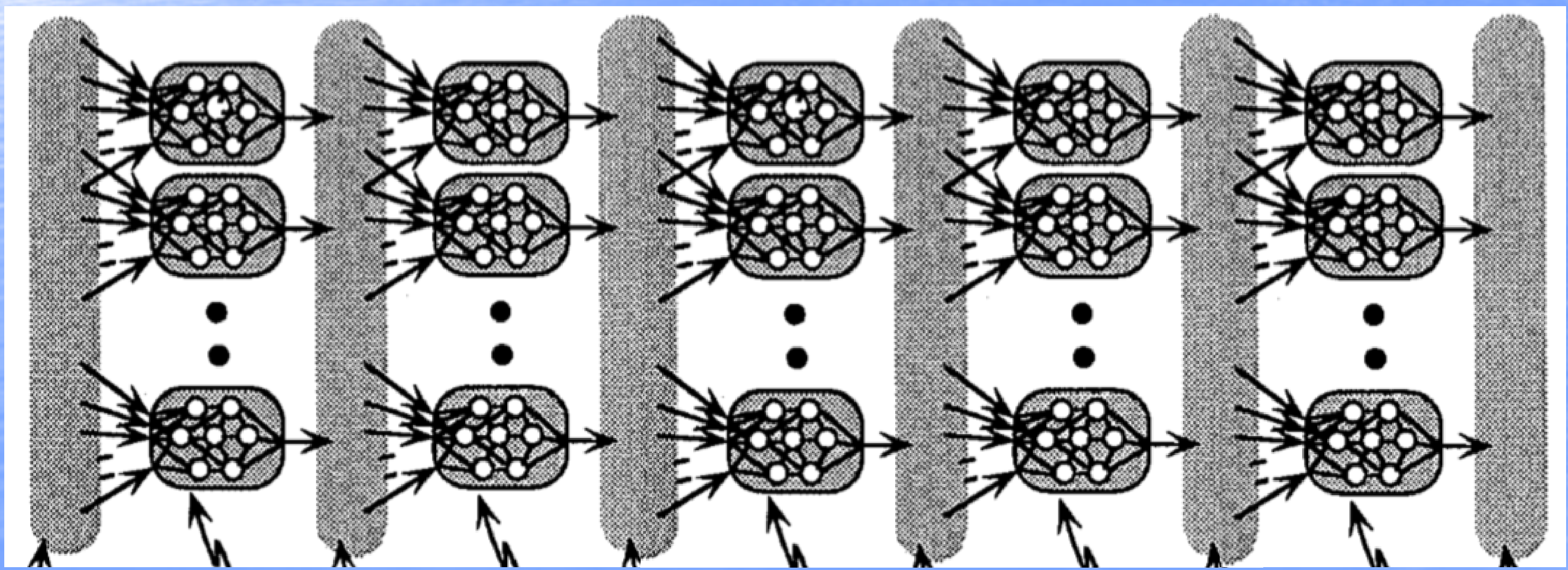
– this type of issue is beyond capability of current ANN technology, but no problem for the brain.

- **Challenge 3:** Black box devices lacking explainability

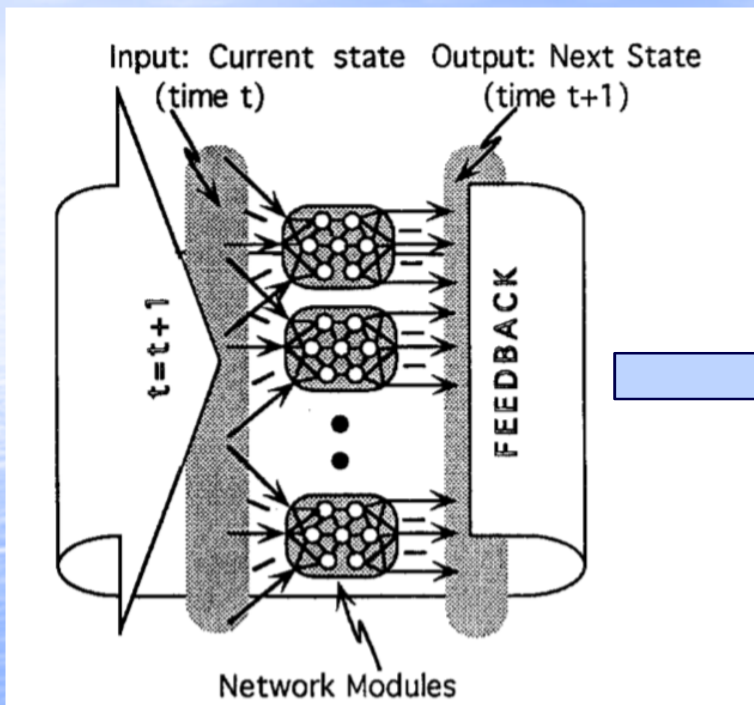


Deep Learning ANNs (DLANNs)

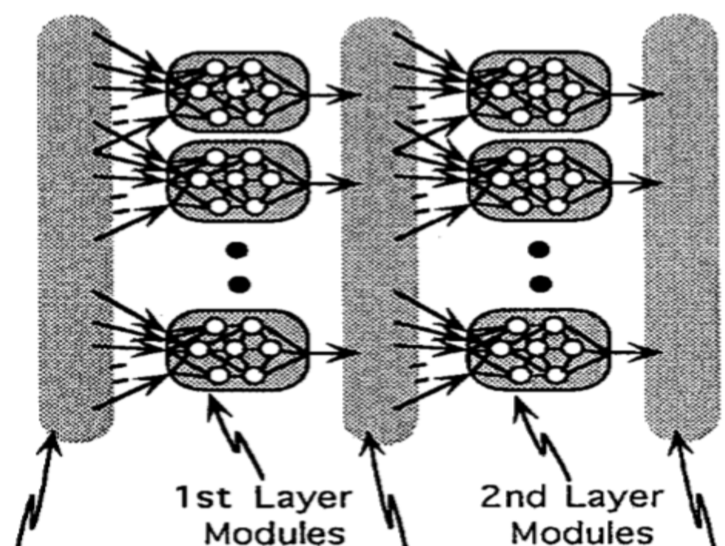
- Main characteristics of DLANNs:
  - Inspired by biological systems (such as primary visual cortex) which process information across a cascade of **many** layers.
  - Successive layers process information from the previous layer.
  - Early layers extract simple features (e.g. boundaries in an image), with later layers identifying progressively more abstract (higher order) features (e.g. lines, shapes, facial features, person, etc…)



- Layers can be:
  - Physically different sets of neurons
  - Unfolded layers from a recurrent architecture



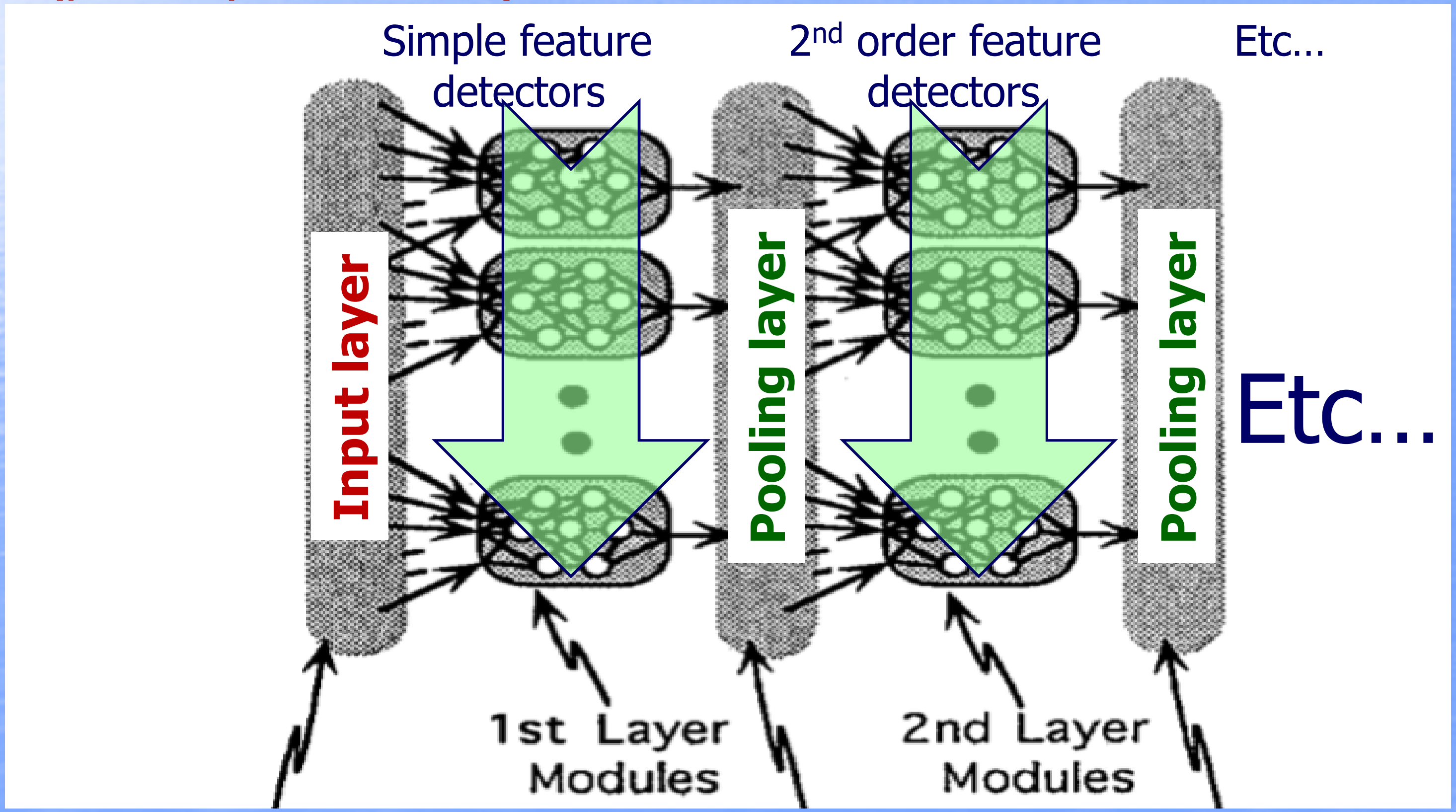
Unfolds to this,…
each layer is architecturally identical



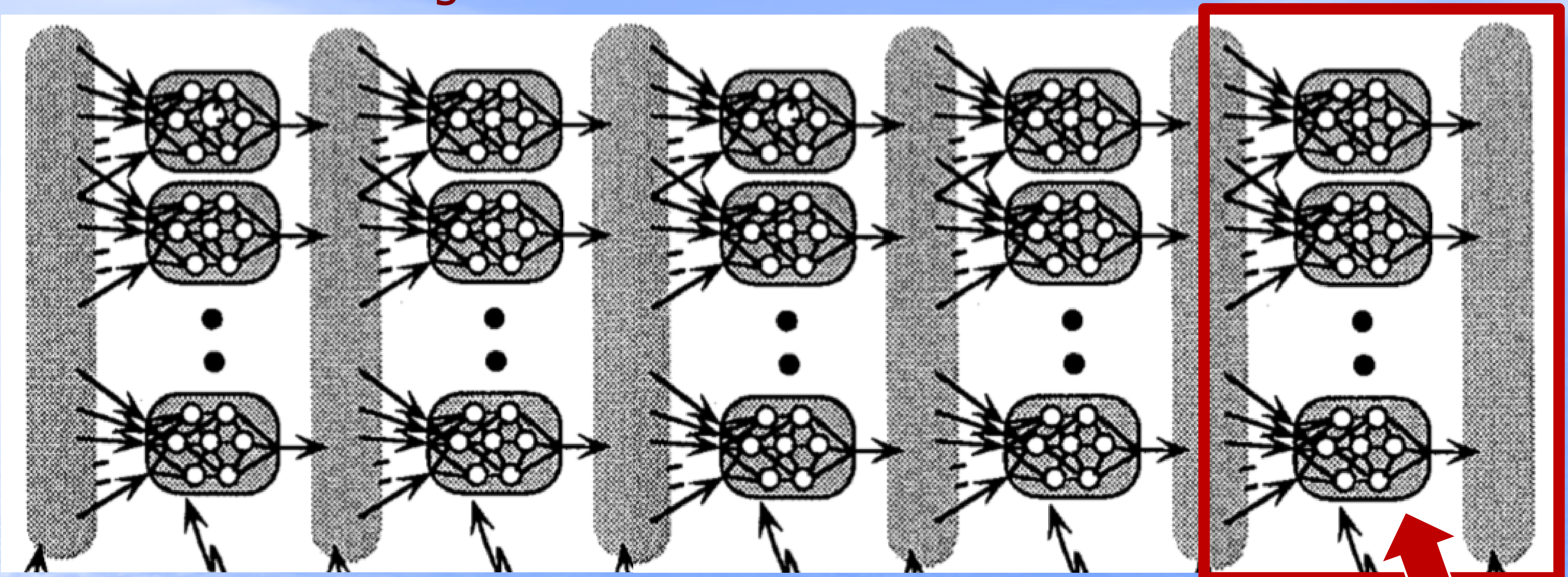
Etc…

- What is the excitement?:
  - DLANNs have been around for decades, their performance has crossed a critical barrier due to a combination of developments.
  - This decade, DLANNs have outperformed other solution methods, including humans, for a range of tasks (pattern recognition, drug analysis, cancer identification…).
  - GPUs have been found well suited to deep learning implementations, reducing processing times by orders of magnitude.

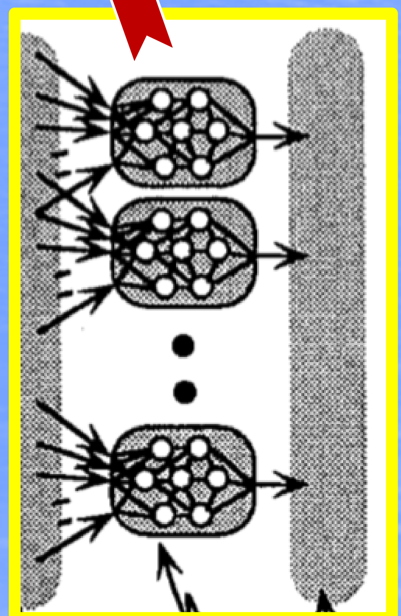
- Example Architecture: Convolutional DLANNs (image processing):
  - Feature detectors scan across the input field
  - In essence, are replicated many times across the field
  - This has the added advantage of making the image size scalable (partially extensible)



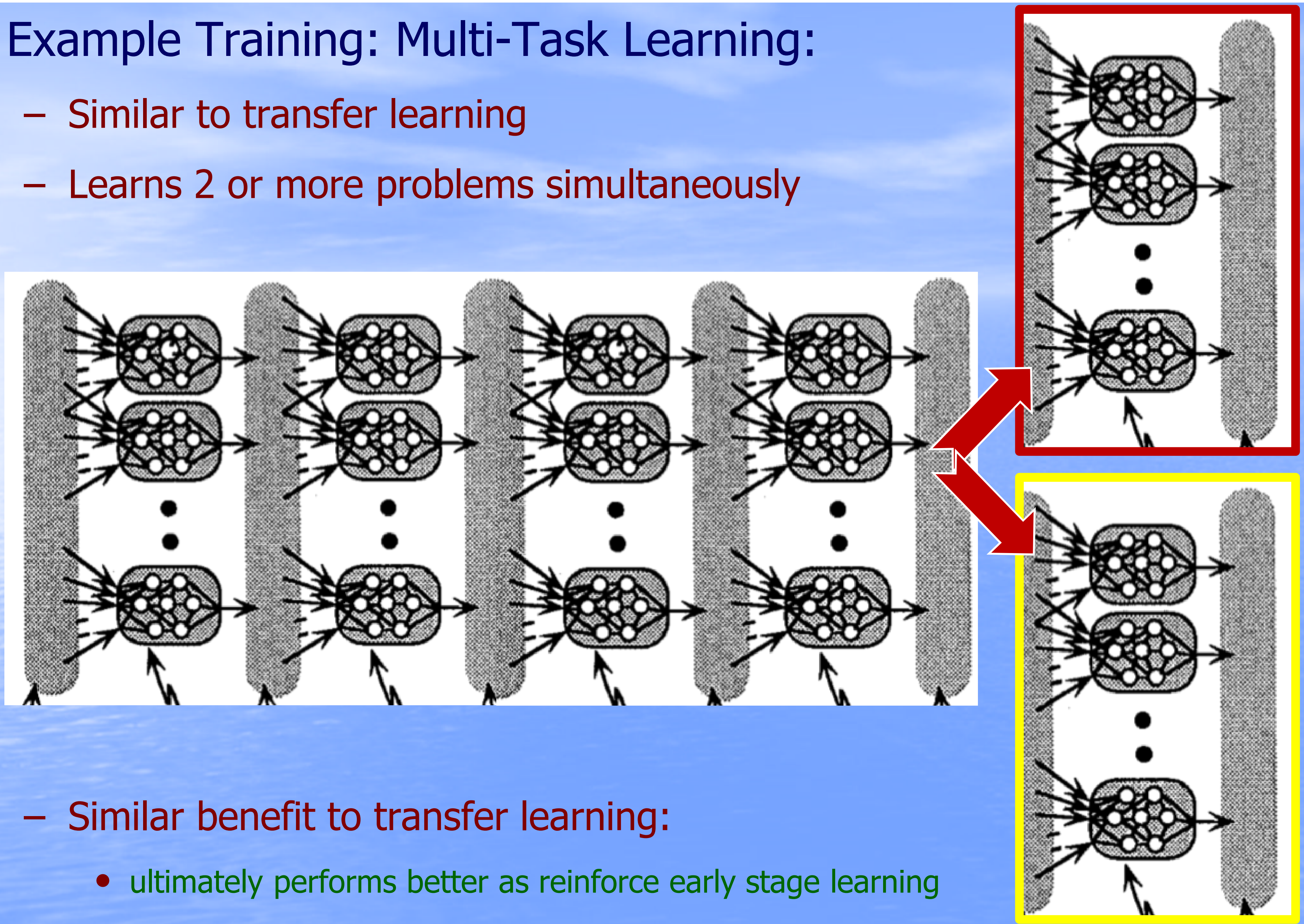
- Example Training: Transfer Learning:
  - Transfer learning



  - For new (similar) problem, replace last section
  - …then retrain, keeping learned earlier stages
  - Idea:
    - starts better,
    - learns faster, and
    - ultimately performs better



- Example Training: Multi-Task Learning:
  - Similar to transfer learning
  - Learns 2 or more problems simultaneously



  - Similar benefit to transfer learning:
    - ultimately performs better as reinforce early stage learning

Next Generation of ANNs

- The biological model (brain) suggests that:
  - greater complexity $\Rightarrow$ greater cognitive skills
  - complexity = f (size, structure)
- However:
  - "brain size" on its own is a poor indicator of cognitive skills:
    - otherwise Sperm Whale = most intelligent species (8 kg vs. 1.3 kg human)
  - (brain size) / (body mass) is also poor indicator:
    - otherwise Shrew = one of the most intelligent species
  - (brain size) / (expected brain size for body mass) encephalization quotient (EQ):
    - humans most intelligent species (7.6 human vs. 4.6 freshwater dolphin)
  - (brain size) – (expected brain size for body mass) gives brain mass available for purposes other than body monitoring and control:
    - humans = most intelligent species
- Possible future:
  - richly structured networks (not just more layers, but more structure laterally and hierarchically, and hierarchical recursion…)
  - richly structured training schemes (learn in stages, not just transfer and multi-task learning…)

- How do you develop massive, richly structured ANNs that solve non-trivial problems?
- Can it be done using a training mechanism?
  - generally these are used to develop weights not ANN structure
  - a few training mechanisms can develop simple structure, eg:
    - Cascade Correlation (number of hidden neurons)
    - Kohonen Networks (connectivity at 1^{st} level)
  - …but not complex structures
- Inspiration from biological models (copy their structure)?
  - researchers have done for the early stages of the visual system but nowhere near complete understanding yet
  - … moreover, most engineering problems don't have biological analogs with ready solutions
- Could turn to simulated evolution (eg: Genetic Algorithms (GA's)) for a solution

- GA’s has been used in engineering to develop ANN’s for many years
  - …but limited to single network units, not complicated structures

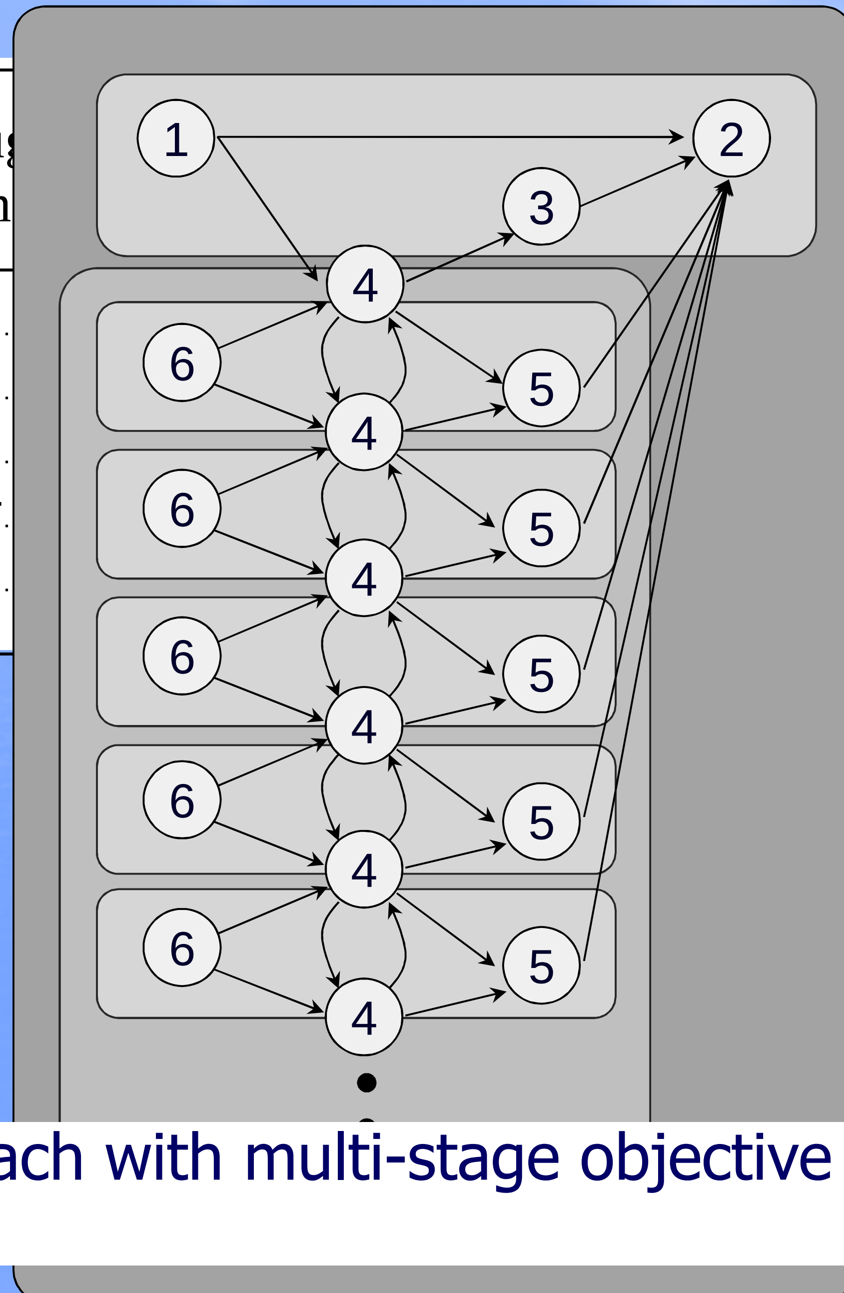
- Special challenges for GA’s in developing massive, richly structured ANNs. Must be able to develop structure at the:
  - macro-level (connectivity between the higher-level units)
  - meso-level (connectivity between neurons within a unit), and
  - micro-level (the mode of operation of the neurons and their links)
  - and do so for very large numbers of neurons (thousands/millions)

- This requires a sophisticated genetic coding system:
  - if millions of neurons, don’t want code with millions of genes
  - ...cumbersome and slow to evolve
  - one possibility is the use of growth algorithms
  - ...simpler codes, especially when repetition in ANN structure:

- Consider the following simple growth table:

Neuron type:	1^{st} daug… neuron…	…er …pe
1 ⇒	2	
2 ⇒		
3 ⇒	5	
4 ⇒	4	
5 ⇒	5	
6 ⇒		

...continue this process:



- could enhance this approach with multi-stage objective functions

APPENDIX: ANN Development Methodology

Common to all ANN development exercises

Step 1: Strategizing

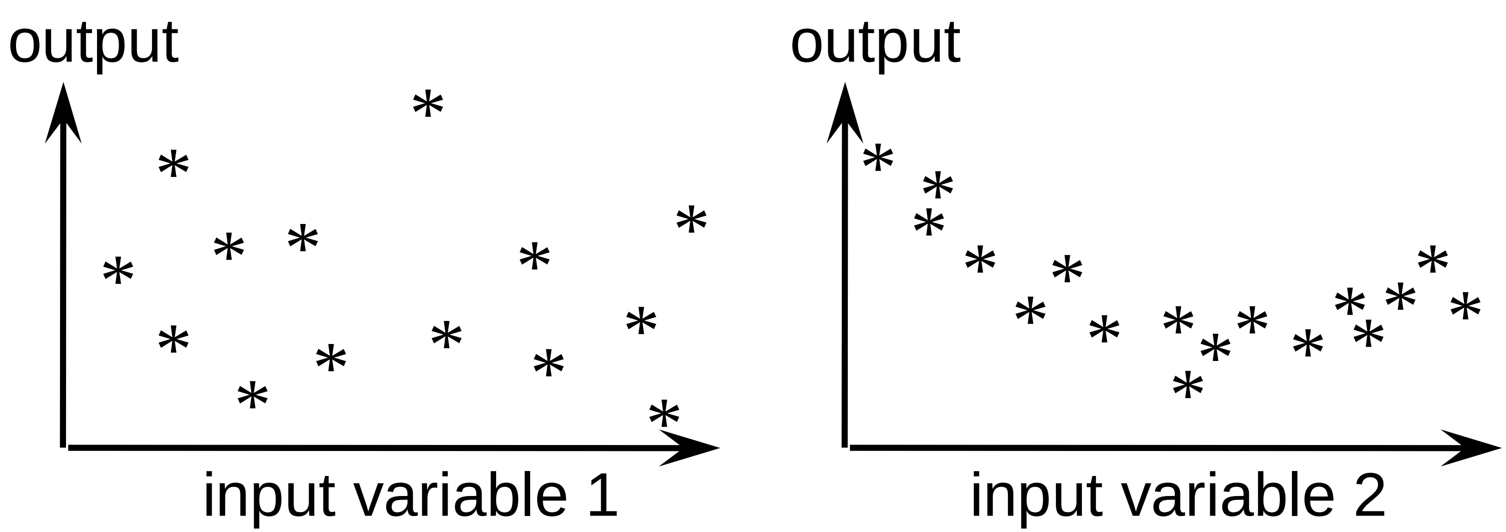
- The aims of strategizing are:
  - Identify the objectives of the study
  - Determine a likely appropriate set of input variables
  - Gain a feel for how the system being modelled responds to different variables, e.g.
    - Linear vs non-linear;
    - Stochastic vs. deterministic, etc…
- Questions to be answered at this stage:
  - What type and structure to adopt for the model?
  - What development algorithm to adopt?
  - What is the objective function?
  - What are the sources for information and what new studies will be required to acquire the necessary data for training, model selection, and validation.
- A pilot study may be required to help answer these questions and to determine feasibility.

Step 1: Strategizing

- Gaining a graphical understanding of the problem can be extremely useful at this stage:
  - Plotting each output variable against each of the input variables:
    - Relevance of each input variable
    - Complexity of the response of the system – e.g. linear vs. non-linear
    - Existence of unexplained variance in the response of the system
  - Plotting each of the input variables against each other
    - Determine correlation between inputs
  - Both approaches illustrated in the following two figures:

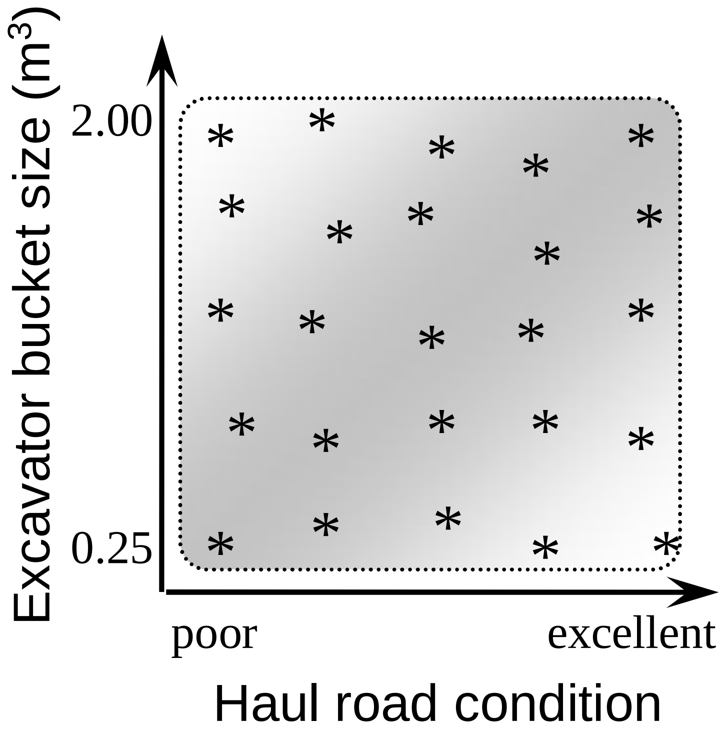
Step 1: Strategizing

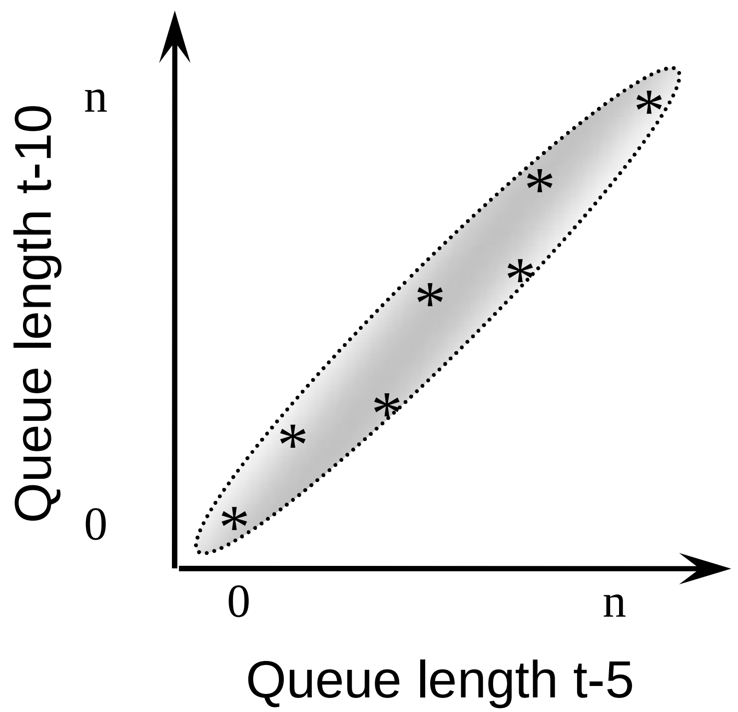
Plotting **Output** vs. **Input** for a Set of Existing Observations of the Response of a System



Step 1: Strategizing

Plotting **Input** vs. **Input** for a Set of Existing Observations of the Response of a System





Uncorrelated

Correlated

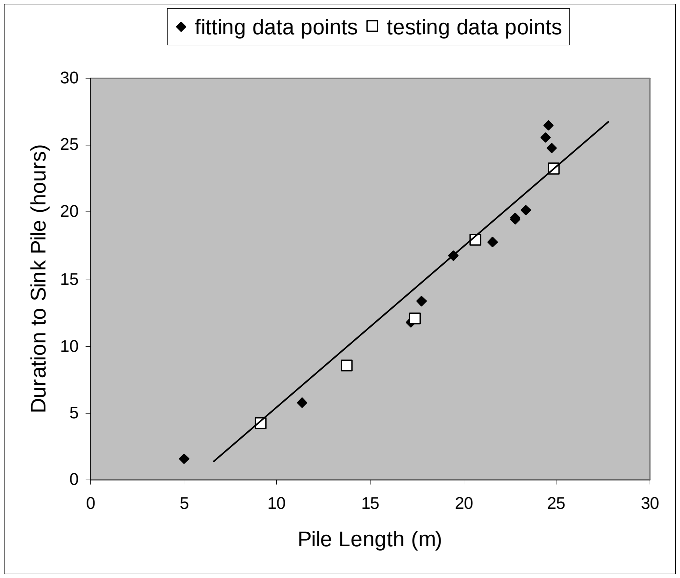
Step 1: Strategizing

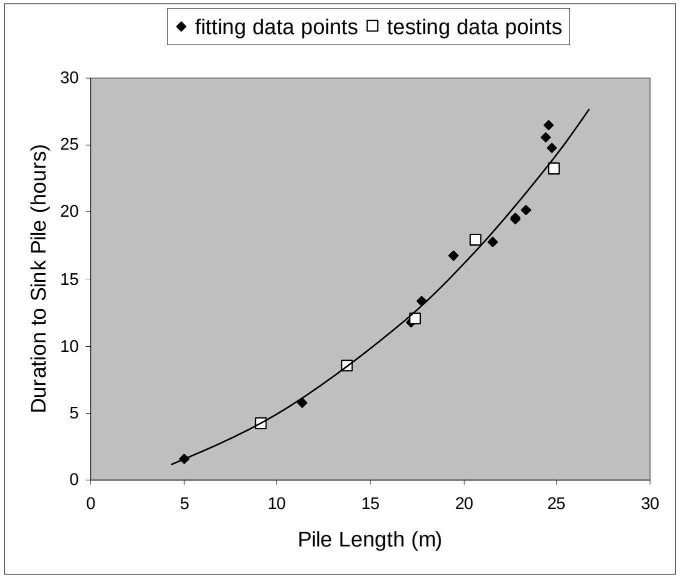
- Understanding a problem is critical to selecting an appropriate type of model:
  - Consider the following:

Step 1: Strategizing

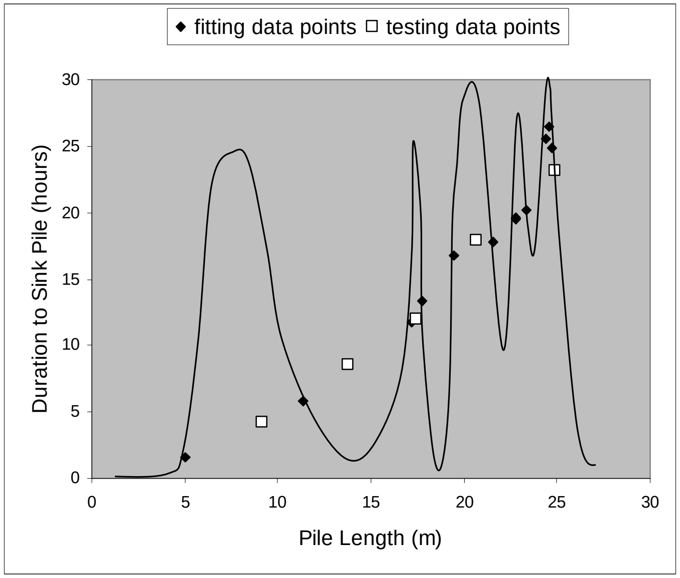
Fitting Functions of Different Complexity to a Set of Observations

Straight Line





Simple Curve



Convoluted Curve
(many degrees of freedom)

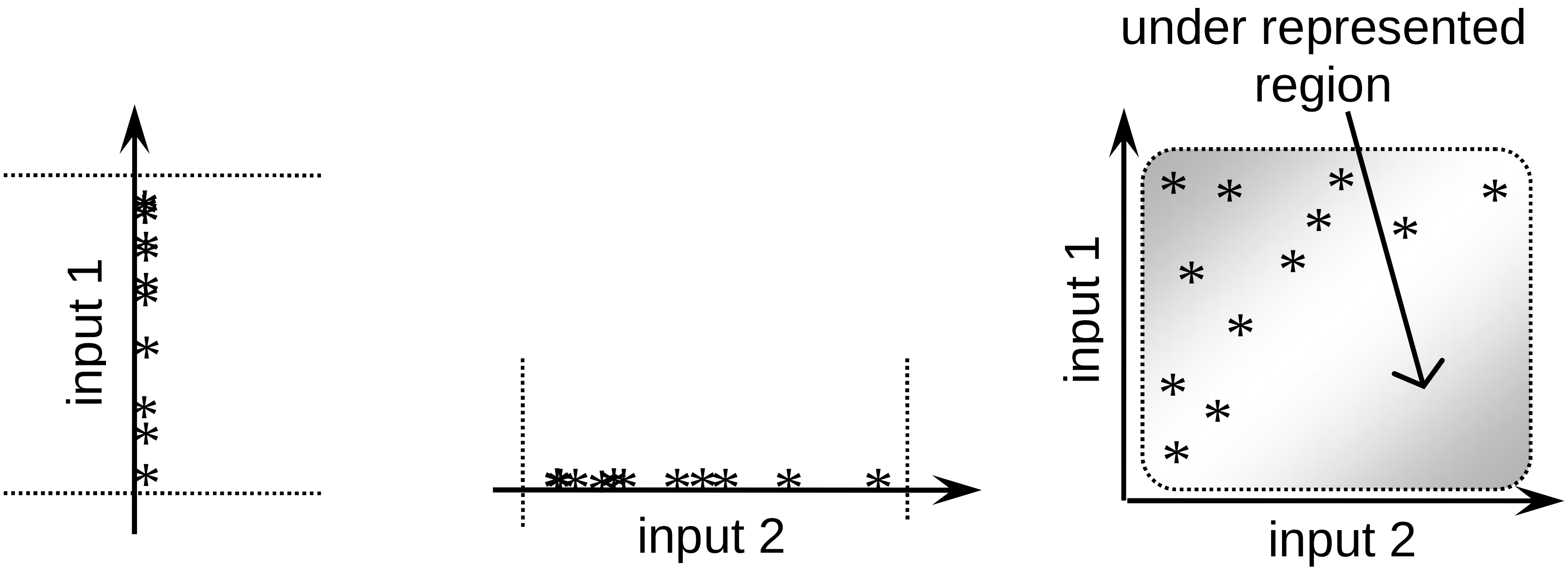
Step 2: Data Collation and Evaluation

- Most empirical modelling studies require 3 sets of data:
  - Training data set – used to develop the model
  - Testing data set – used to compare the performance of alternative models and variants of the model
  - Validation data set – used to make a final validation of the performance of the final model
- Each of these data sets must be assessed or designed to make sure that it is representative of the problem.
- An appropriate data set **size** is dependent on:
  - complexity of the problem…
  - …and may be determined through sensitivity analyses
- An appropriate data set **distribution** is dependent on:
  - form of the problem (some areas may require higher density of observations)…
  - …and may be assessed using graphical plots:

Step 2: Data Collation and Evaluation

Distribution of 12 Observations Across the Problem Domain

Key: [dotted box] = problem domain; * = observation data points



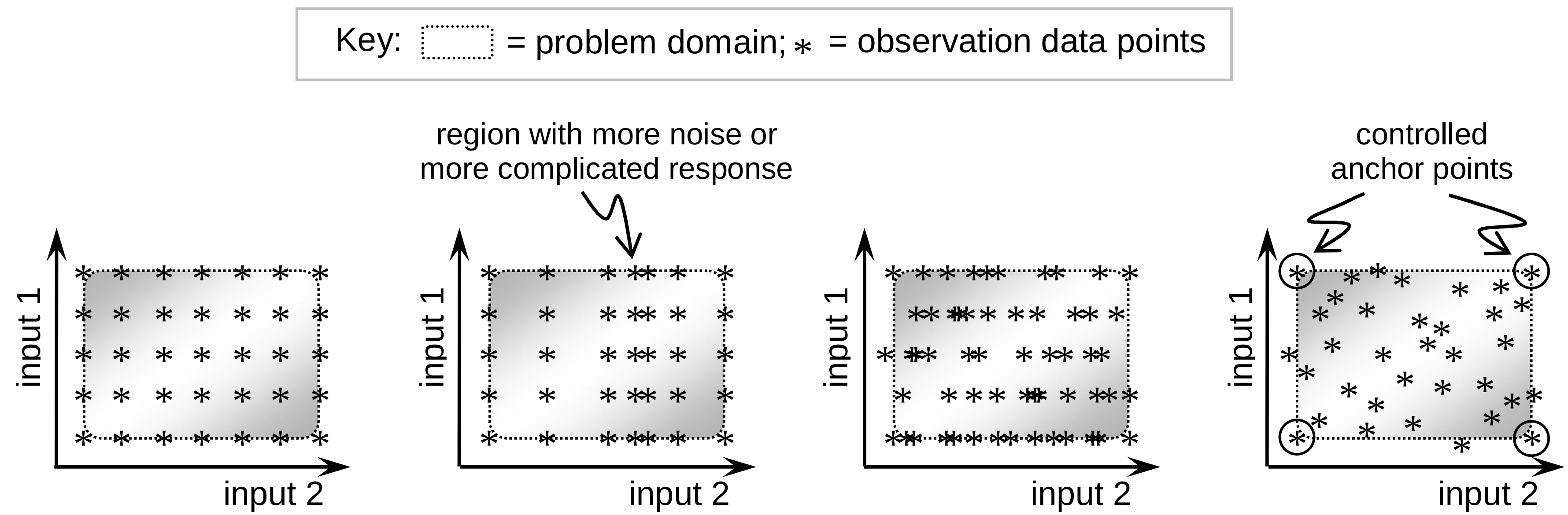
Input 1 | Input 2 | Input 1 vs. Input 2

Step 2: Data Collation and Evaluation

- Where you can control the set of observations used for modelling:
  - Make sure all observations cover the entire problem domain
  - Many layout schemes are available, but make sure appropriate for the problem at hand
  - If use a regular grid, the testing and validation sets should normally still be randomly positioned

Step 2: Data Collation and Evaluation

Distribution of Observations Collected from Controllable Systems



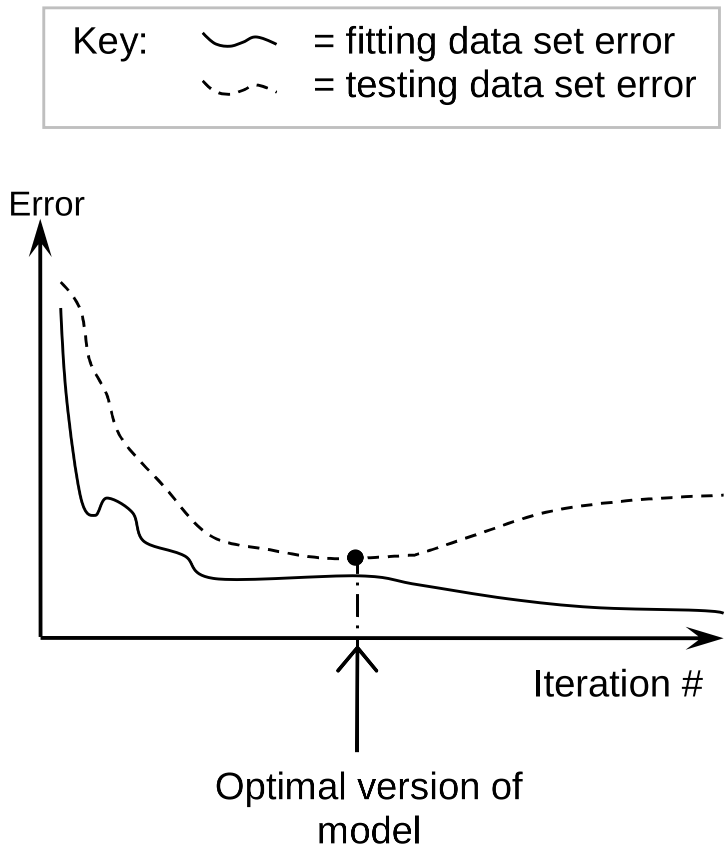
Regular grid. Variable density grid. Regular & random. Random.

Step 3: Model Development

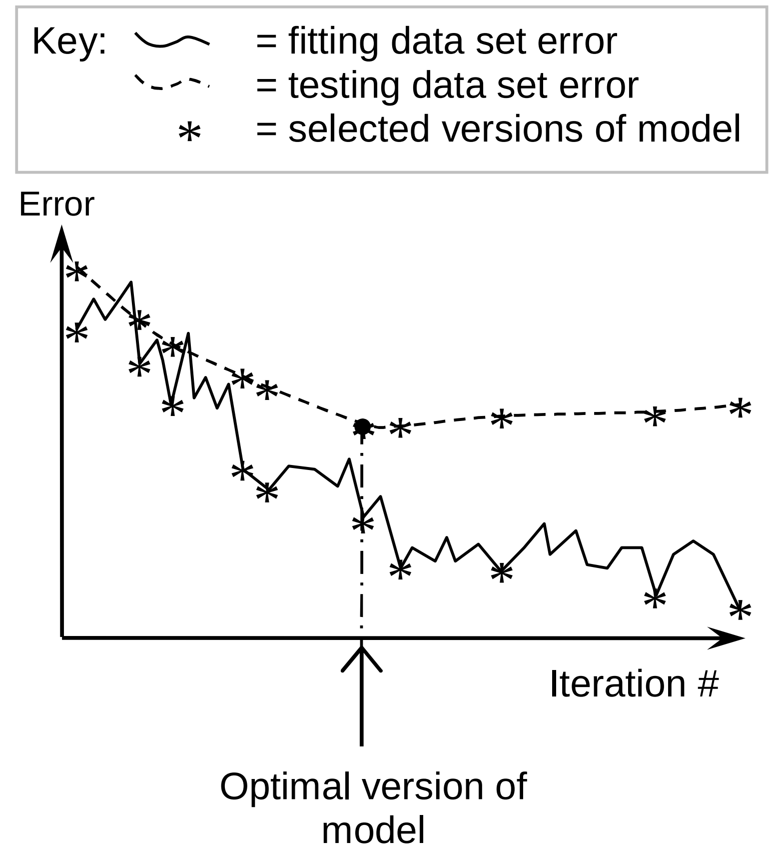
- Whereas step 1 (strategizing) identified a conceptual design for the model,…
- …step 3 develops the finalized design for the model.
- Progress in training can be monitored for both the training data set and the testing data set:
  - Training terminates where the testing data set performs optimally…
  - …going beyond this point can cause 'overtraining' (memorization);
  - consider the following:

Step 3: Model Development

Progress in Model Development for Studies that use Search Algorithms



Error gradient descent applied to model coefficients.



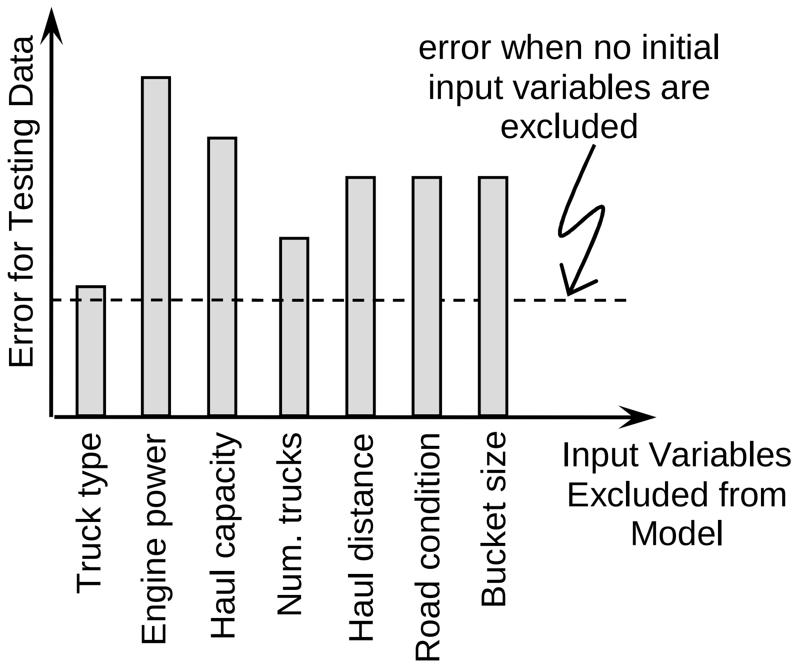
Simulated evolution applied to model structure.

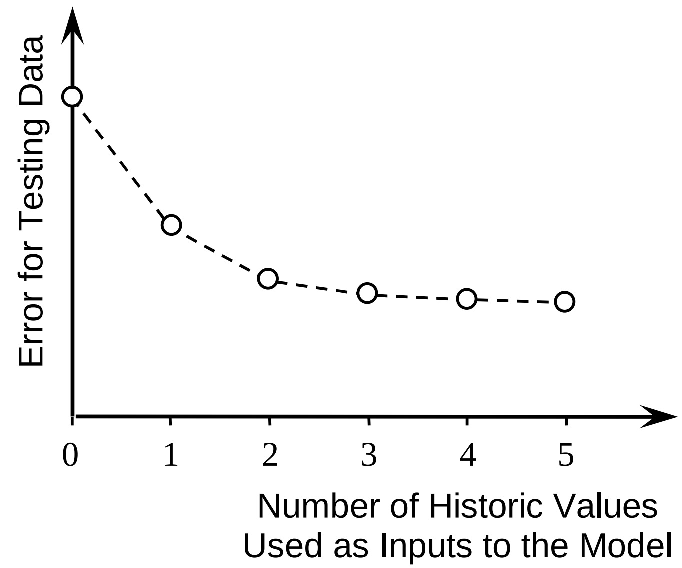
Step 3: Model Development

- Some model parameters are not adjusted by the model development/training algorithm, e.g.:
  - Number of layers in a neural net
  - Number of neurons in a layer of a neural net
  - Number of observations used for training
  - Set of input variables used, etc…
- These will need to be adjusted manually, and in a methodical way:

Step 3: Model Development

Searching for an Input Configuration for a Model (Excavation) that Minimizes the Testing Error





Alternative sets of input variables.

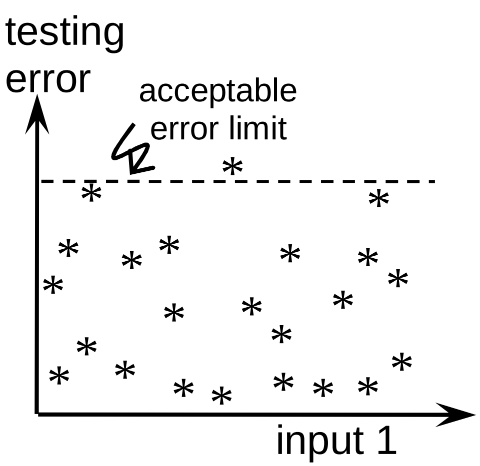
Alternative numbers of historic input values.

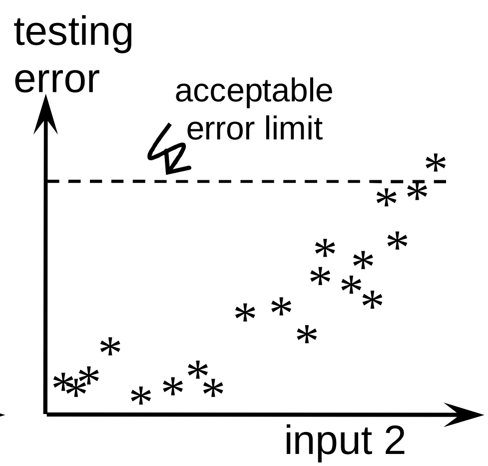
Step 4: Model Evaluation and Final Selection

- The study at this stage may have generated several candidate models
- These should be thoroughly evaluated using the **testing data set** to select the best
- Performance should not be based just on the objective function…
- …the performance across the problem domain should also be considered to look for consistency in performance:

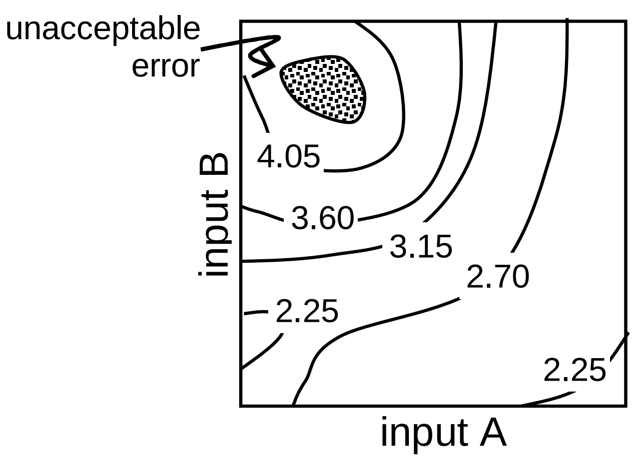
Step 4: Model Evaluation and Final Selection

Evaluating Error across the Problem Domain





Error plotted against input variable.



Error plotted as a contour map.

Step 5: Final Validation

- At this stage we have the final version of the model
- This needs to be validated:
  - to get an accurate assessment of its performance
  - to see whether further development may be required
- Should not use the testing data set for this as the model may have some bias towards it
- Requires a 3rd independent data set.

Step 6: Implementation and Review

- Education of end-users:
  - Collection and organization of input data to ensure model validity
  - Interpretation of the output from the model
  - Usage of the model for problem solving
- Where possible, feedback from use to continue validation and improvement of the model.

Bibliography:

- Goodfellow, I., Bengio, Y., and Courville, A., (2016). *Deep Learning*, MIT Press.
- Schmidhuber, J. (2015). "Deep Learning in Neural Networks: An Overview". *Neural Networks*. 61: 85–117.
- Wang, Y., Flood, I., and Issa, RRA., (2015). "Comparison of Artificial Neural Networks and Support Vector Machines for Weigh-In-Motion Based Truck Type Classification", *2015 International Symposium on Empirical Modeling, INFOCOMP 2015*, IARIA, June 21-26, Brussels, Belgium, 6 pp.
- Bengio, Y., LeCun, Y., and Hinton, G., (2015). "Deep Learning", *Nature*. 521 (7553): 436–444.
- Flood, I, Bewick, BT, and Rauch, E.*, (July 2011). "Rapid Simulation of Blast Wave Propagation in Built Environments using Coarse-Grain Based Intelligent Modeling Methods", *18th EG-ICE Workshop, European Group for Intelligent Computing in Engineering*, Twente, The Netherlands, 9 pp.
- Bewick, BT., Flood, I., and Chen, Z., (2011). "A Neural-Network Model-Based Engineering Tool for Blast Wall Protection of Structures", *International Journal of Impact Engineering*, Multi-Science Publishing, London, 2 (2), pp 169-176.
- Flood, I., and Issa, RAA., "Empirical Modeling Methodologies for Construction", in Journal of Construction Engineering and Management, ASCE, New York, Vol. 36, No. 1, (2010), pp 36-48.
- Flood, I., Bewick BT., Dinan, RJ., and Salim, HA., (2009). "Modeling Blast-Wave Propagation using Artificial Neural Network Methods", *in International Journal of Advanced Engineering Informatics*, Elsevier, Amsterdam, Vol. 23, No. 4, pp 418-423.
- Flood, I., (2008). "Towards the Next Generation of Artificial Neural Networks for Civil Engineering", *in Advanced Engineering Informatics*, Vol. 22, No. 1, Elsevier, Amsterdam, pp 4-14.
- Flood, I. and Kartam, N., "Neural Networks in Civil Engineering. I: Principles and Understanding", in Journal of Computing in Civil Engineering, ASCE, Vol. 8, No. 2, (April 1994), pp 131-148.
- Flood, I. and Kartam, N., "Neural Networks in Civil Engineering. II: Systems and Application", in Journal of Computing in Civil Engineering, ASCE, Vol. 8, No. 2, (April 1994), pp 149-162.
- Gagarin, N.*, Flood, I. and Albrecht, P., "Computing Truck Attributes with Artificial Neural Networks", in Journal of Computing in Civil Engineering, ASCE, Vol. 8, No. 2, (April 1994), pp 179200.