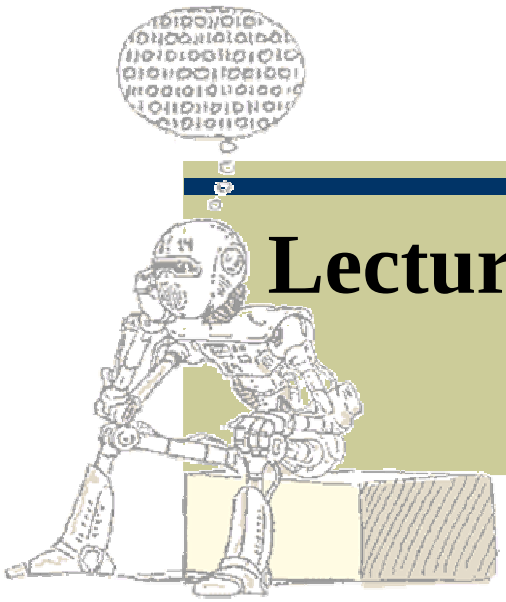


Lecture XI– Learning with Distal Teachers (Forward and Inverse Models)

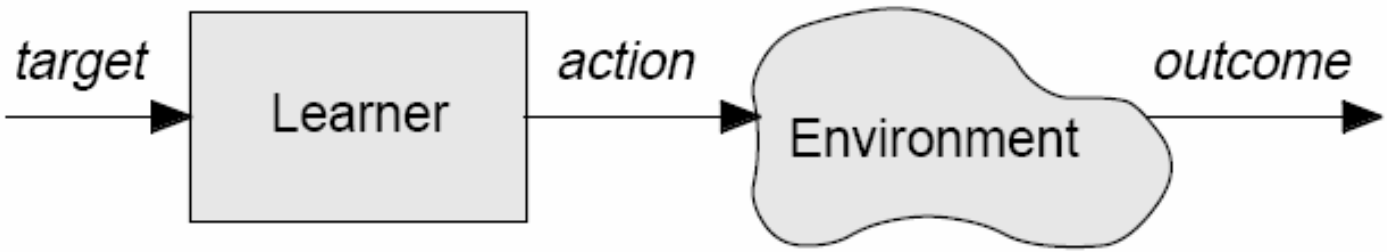
Contents:

- The Distal Teacher Problem
- Distal Supervised Learning
- Direct Inverse Modeling
- Forward Models
- Jacobian
- Combined Inverse & Forward Models

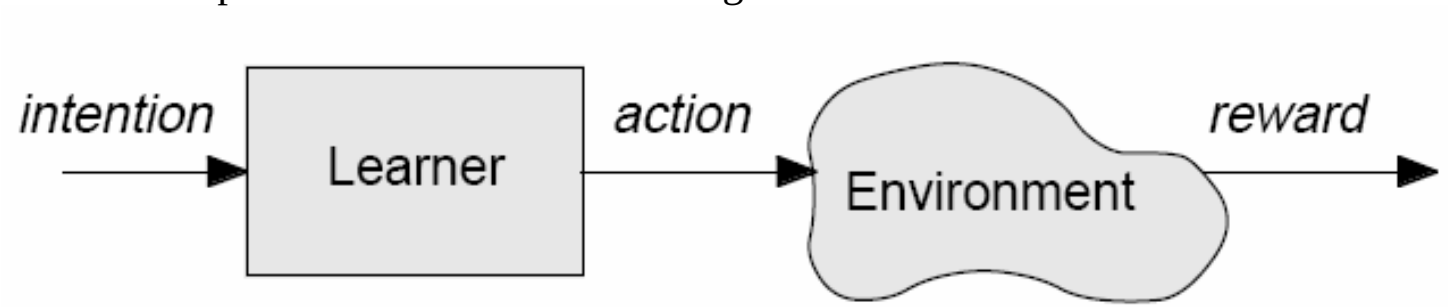
The Distal Teacher Problem

For certain problems, i.e., learning to act, teaching signals are not easily obtained:

- Example 1: Distal Supervised Learning



- Example 2: Reinforcement Learning



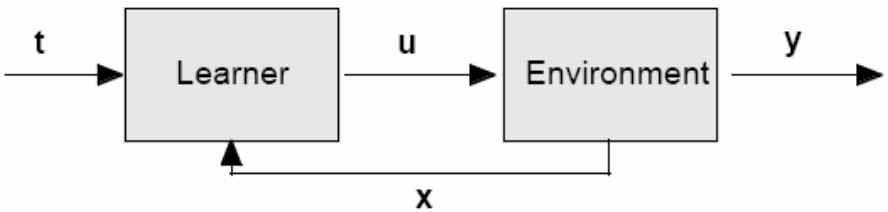
How to learn to choose the correct action without getting supervised information about the quality of an action?

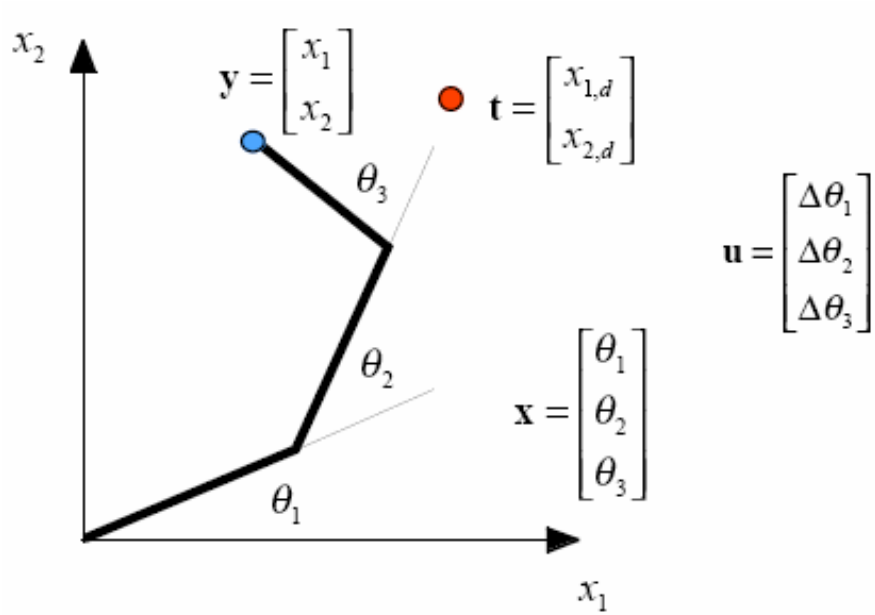
Distal Supervised Learning

- **Given:**
  - target **t** in distal space and the current state of the system **x**
  - an unknown system **y**=f(**x**,**u**)
- **Goal:**
  - find the action **u** to achieve **t**

x and **y** are not the same: **y** is a certain observable as the outcome of action **u** in state **x**

Example:

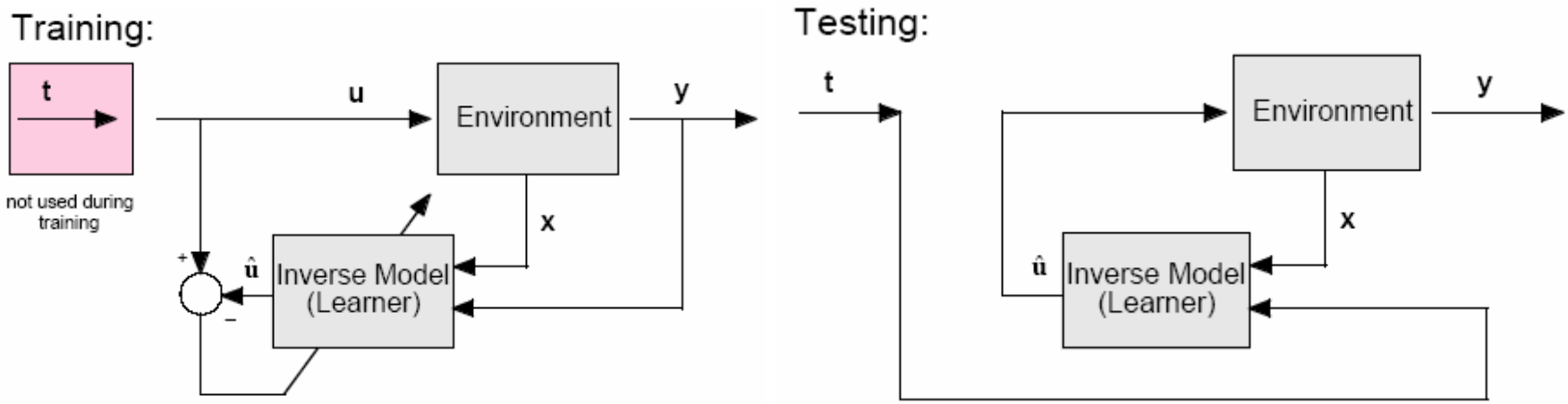




Direct Inverse Models

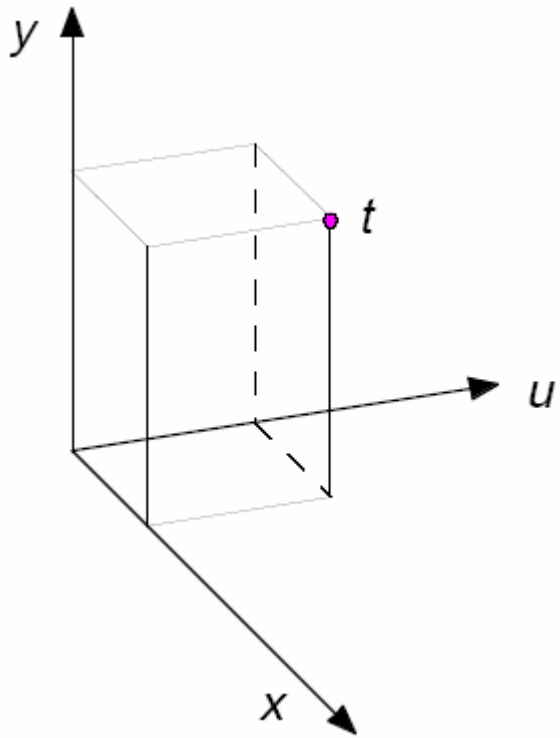
- **Definitions:**
  - Forward Model $\mathbf{y}=f(\mathbf{x})$
  - Inverse Model $\mathbf{x}=f^{-1}(\mathbf{y})$

- **Example:**



Problems of Direct Inverse Learning: Training Data

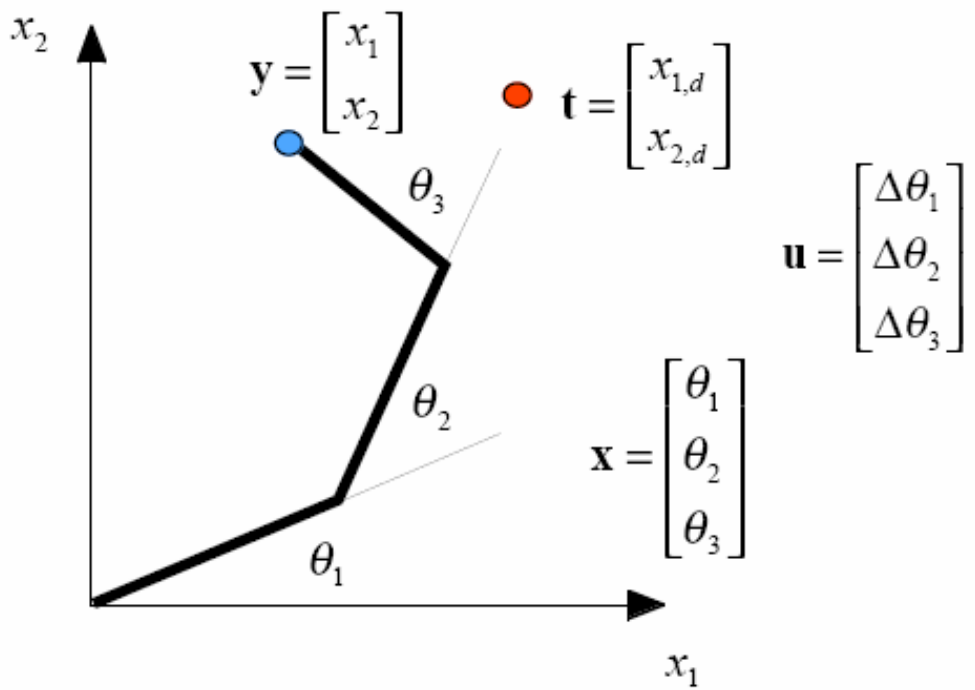
- **How to generate useful training data?**
  - exhaustive data generation
    - only possible for low dimensional problems (the curse ...)
    - for real physical systems, it is not easy to drive the system to generate exhaustive data
  - use incompletely trained model to generate "explorative" data
    - predict **u** for targets **t**
    - then apply those predicted **u**, observe **y**, update the learning system
    - but there is a big danger of getting stuck in irrelevant training data!



Problems of Direct Inverse Learning: Non-Uniqueness

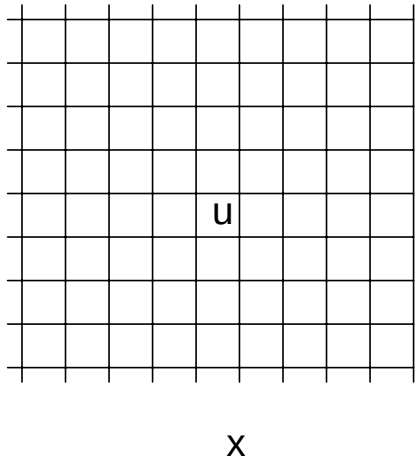
- What if the Inverse Function is not unique?
  - Example 1: $y = au_1 + bu_2 + cx$
  - Example 2: how to determine **u** from just two equations, but 3 unknowns?

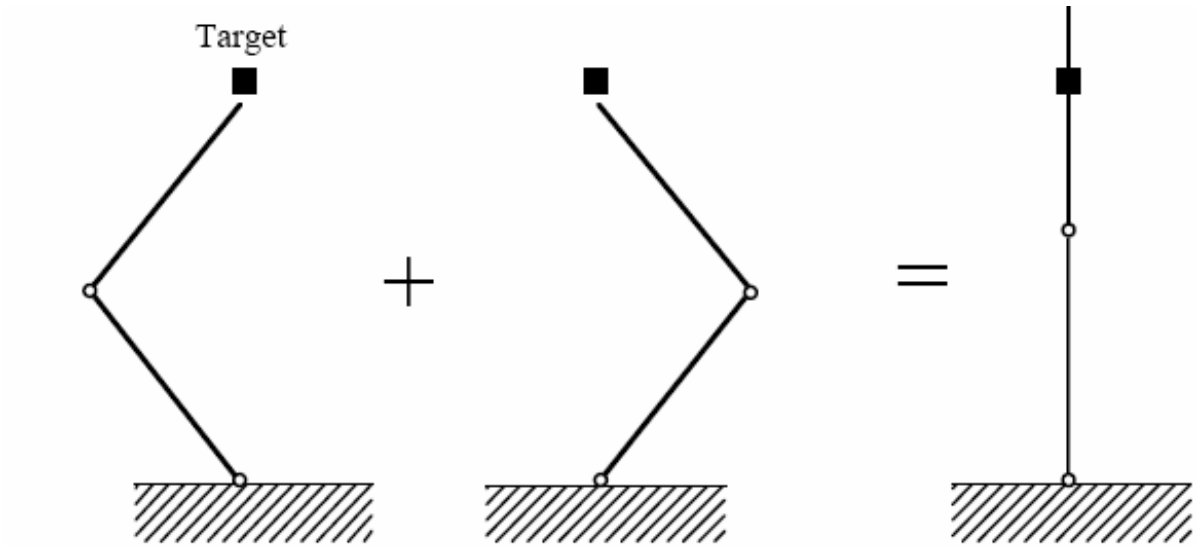
Important: For non-unique inverse models, the learning system has to pick ONE out of many solutions, but it also has to ensure that this solution is REALLY a valid solution!



Can we learn Non-Unique mappings?

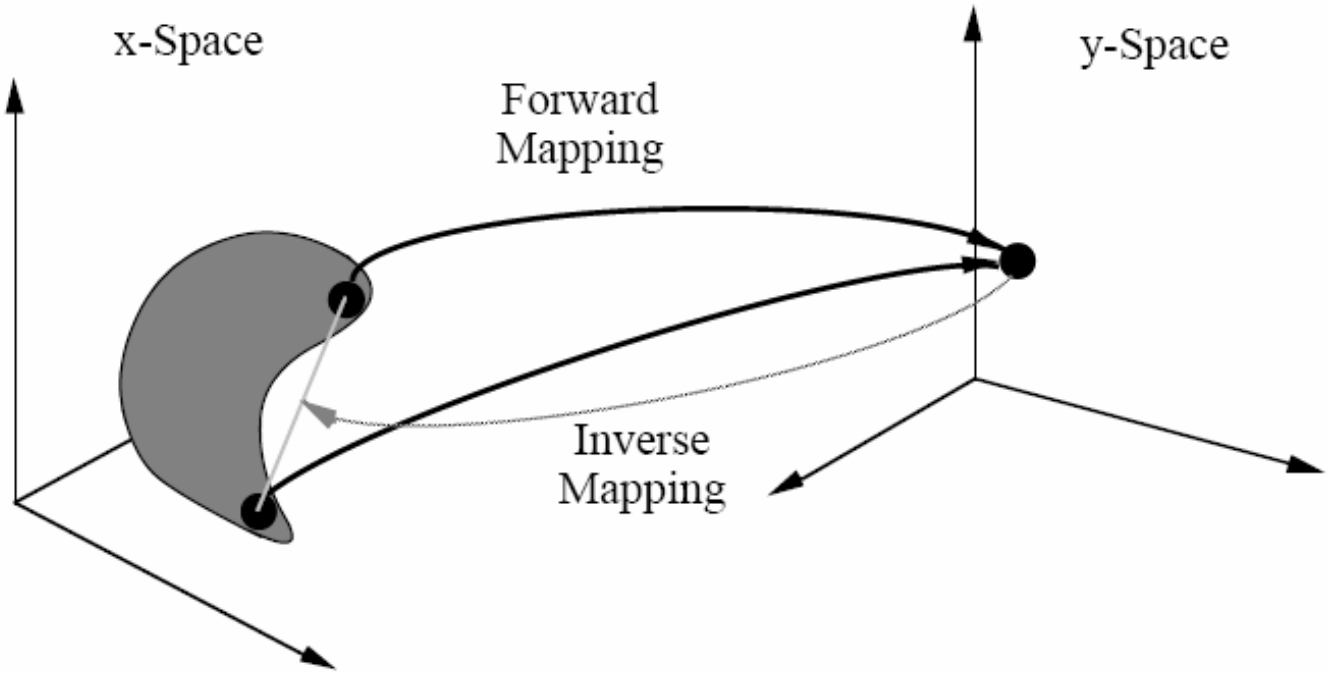
- **Analogy: Learning networks are like lookup-tables**
  - they average over all outputs that belong to the same input in order to eliminate noise!





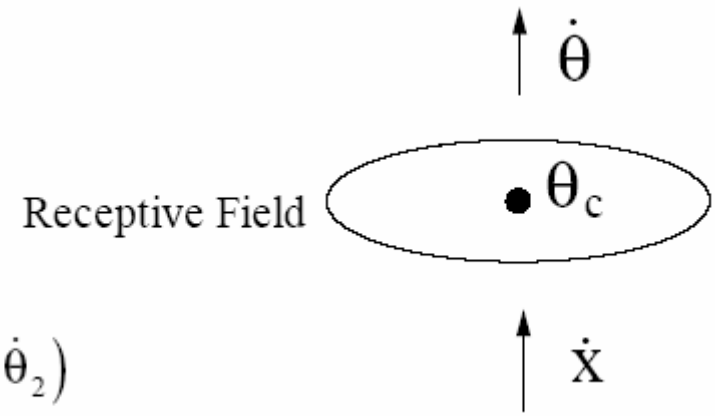
Convex and Non-convex Mappings

- **When can we learn a non-unique inverse model?**
  - Only when the mapping is convex in output space!



Dealing with Non-Convex Inverse Models

- **Only Use Training Data from a Convex Region**
  - this is usually very hard
- **Change of Representation in Conjunction with Spatially Localized Learning**
  - this is REALLY cool if applicable!



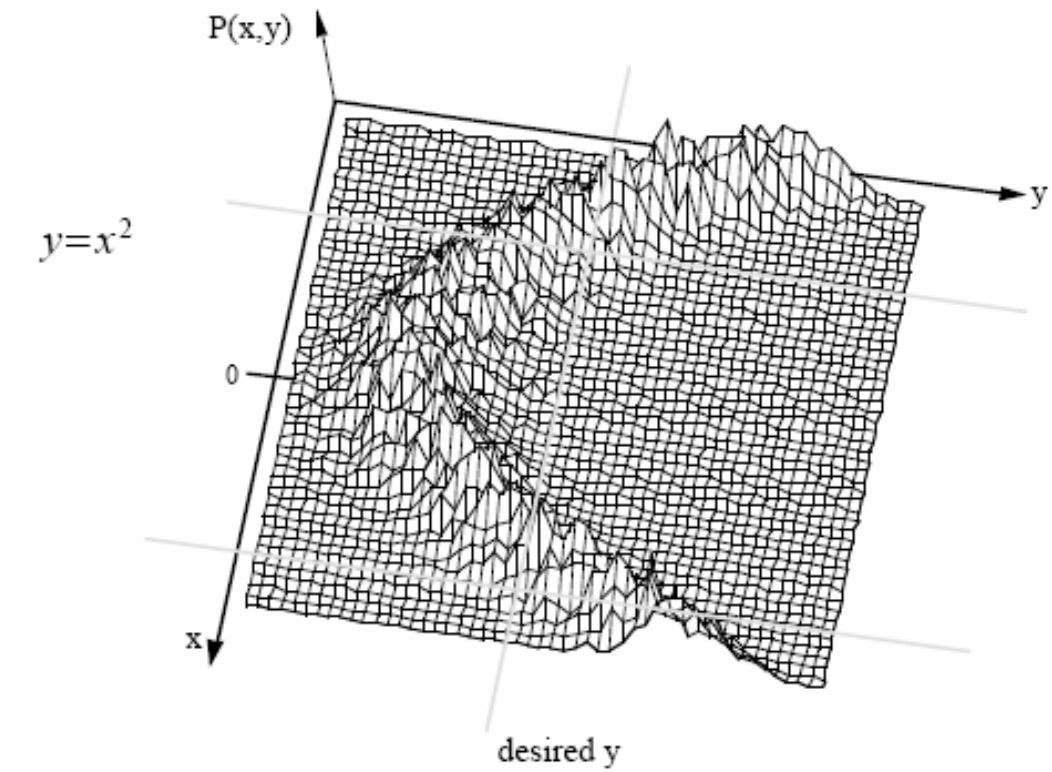
$$\mathbf{x} = f(\theta) \quad \Rightarrow \quad \dot{\mathbf{x}} = J(\theta)\dot{\theta}$$

$$\dot{\mathbf{x}} = J(\theta)\dot{\theta}_1; \quad \dot{\mathbf{x}} = J(\theta)\dot{\theta}_2 \quad \Rightarrow \quad \dot{\mathbf{x}} = J(\theta)\left(\dot{\theta}_1 + \dot{\theta}_2\right)$$

$$\Rightarrow \quad (\dot{\mathbf{x}}, \theta) \rightarrow \left(\dot{\theta}\right) \text{ is a convex mapping}$$

Dealing with Non-Convex Inverse Models

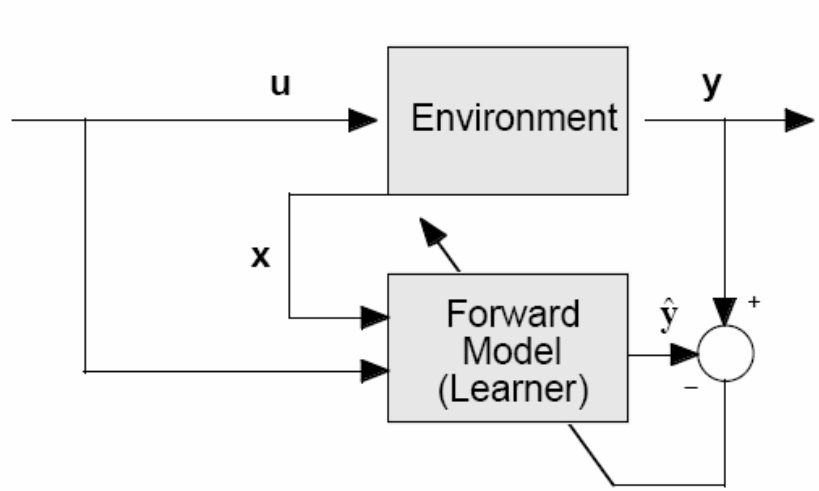
- **Learning the Joint Density**
  - There are various Bayesian and nonparametric methods to do this



Dealing with Non-Convex Inverse Models

- **Search in Forward Model**
  - Learn (well-define) forward model
  - Search for appropriate action by gradient descent (a more AI-ish technique)
    - this will generate a new action that should give interesting new training data that helps to achieve the target

$$\mathbf{u}^{n+1} = \mathbf{u}^{n} - \alpha \frac{\partial J}{\partial \mathbf{u}}, \text{ where } J = (\mathbf{t} - \mathbf{y})^{T}(\mathbf{t} - \mathbf{y})$$



The Jacobian

- **Taking derivatives through neural networks**
  - From least squares learning, we already know the derivative:

$$\frac{\partial J}{\partial \mathbf{w}} = -\frac{\partial \mathbf{y}}{\partial \mathbf{w}}^{T} (\mathbf{t} - \mathbf{y}) \text{ where } J = (\mathbf{t} - \mathbf{y})^{T} (\mathbf{t} - \mathbf{y})$$

  - But we can the same easily calculate the derivative with respect to any other quantity of the network, e.g., with respect to the input

$$\frac{\partial J}{\partial \mathbf{x}} = -\frac{\partial \mathbf{y}}{\partial \mathbf{x}}^{T} (\mathbf{t} - \mathbf{y})$$

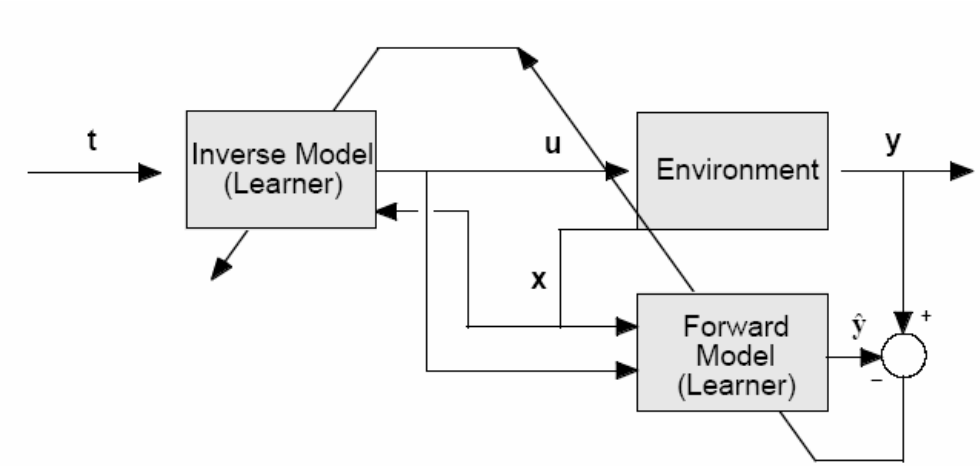
The derivative of the outputs with respect to the inputs $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}^{T}$

is called the **Jacobian Matrix**. Note that the coefficients of the matrix change nonlinearly as function of the **x** and the **y** data. The derivation of this derivative is just like backprop or gradient descent in RBFs. (Bishop, Ch.4)

Inverse-Forward Model Combination

- Instead of using search techniques, neural nets can learn the entire distal teacher problem directly

$$\frac{\partial J}{\partial \mathbf{w}_{inv}} = -\frac{\partial \mathbf{y}}{\partial \mathbf{w}_{inv}}^{T}(\mathbf{t}-\mathbf{y}) = -\frac{\partial \mathbf{u}}{\partial \mathbf{w}_{inv}}^{T}\frac{\partial \mathbf{y}}{\partial \mathbf{u}}^{T}(\mathbf{t}-\mathbf{y})$$



Inverse-Forward Model Combination

- What is the correct error signal to use for learning:
  - **the forward model**

    $(\mathbf{t} - \mathbf{y})$ or $(\mathbf{t} - \hat{\mathbf{y}})$ or $(\mathbf{y} - \hat{\mathbf{y}})$
  - **the inverse model?**

    $(\mathbf{t} - \mathbf{y})$ or $(\mathbf{t} - \hat{\mathbf{y}})$ or $(\mathbf{y} - \hat{\mathbf{y}})$

- How accurate does the Forward Model have to be to learn an accurate Inverse Model?
- Which solution to the inverse problem does the Learning box acquire?

We have as error criteria:
the prediction error, the performance error, the predicted performance error

Learning the forward model: use the prediction error, this is clear!
Learning the Inverse model: the performance error or the predicted performance error can be used.

Inverse-Forward Model Combination

- Additional Optimization Criteria to Select a Particular Inverse Solution

$$e.g.: J = \frac{1}{2}(\mathbf{t} - \mathbf{y})^T(\mathbf{t} - \mathbf{y}) + \gamma \frac{1}{2}\mathbf{u}^T\mathbf{u}$$

$$\frac{\partial J}{\partial \mathbf{w}_{inv}} = -\frac{\partial \mathbf{y}}{\partial \mathbf{w}_{inv}}^T(\mathbf{t} - \mathbf{y}) + \gamma \frac{\partial \mathbf{u}}{\partial \mathbf{w}_{inv}}^T \mathbf{u}$$

$$= -\frac{\partial \mathbf{u}}{\partial \mathbf{w}_{inv}}^T \frac{\partial \mathbf{y}}{\partial \mathbf{u}}^T(\mathbf{t} - \mathbf{y}) + \gamma \frac{\partial \mathbf{u}}{\partial \mathbf{w}_{inv}}^T \mathbf{u}$$

$$= -\frac{\partial \mathbf{u}}{\partial \mathbf{w}_{inv}}^T \left(\frac{\partial \mathbf{y}}{\partial \mathbf{u}}^T(\mathbf{t} - \mathbf{y}) - \gamma \mathbf{u} \right)$$

Summary

- Certain problems cannot just be learned—they are theoretically and practically impossible to learn (e.g., non-convex inverses)
- Clever representations and network architectures allow to overcome such problems (after the essence of the problem has been understood)
- distal supervised learning is a powerful method to approach inverse problems and simple reinforcement learning problems