Parametric Models Fitting with Automatic Differentiation

Gabriel Peyré

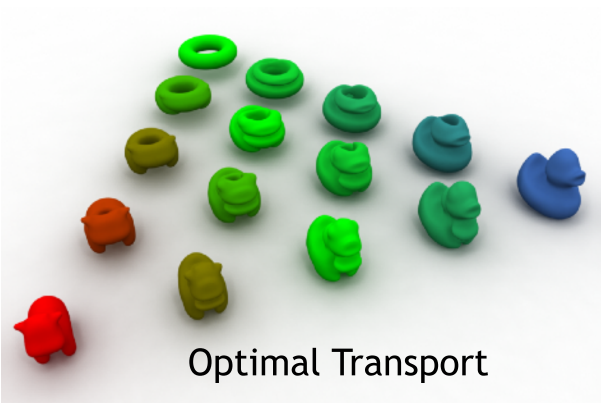
www.numerical-tours.com

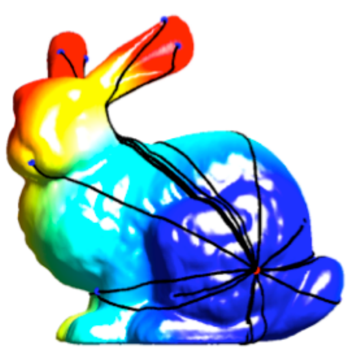
Mathematical Coffees

Huawei-FSMP joint seminars

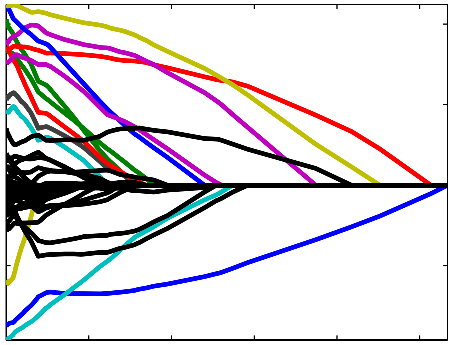
https://mathematical-coffees.github.io

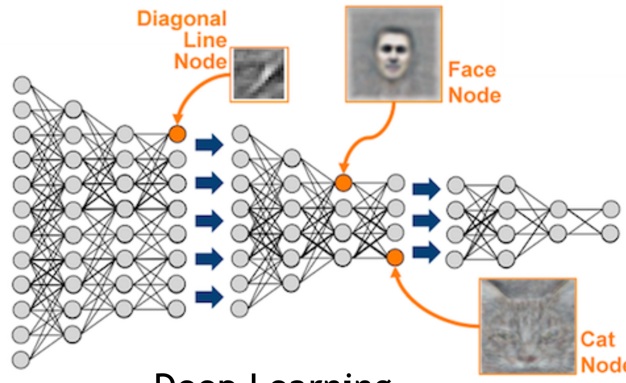
Organized by: Mérouane Debbah & Gabriel Peyré


Optimal Transport

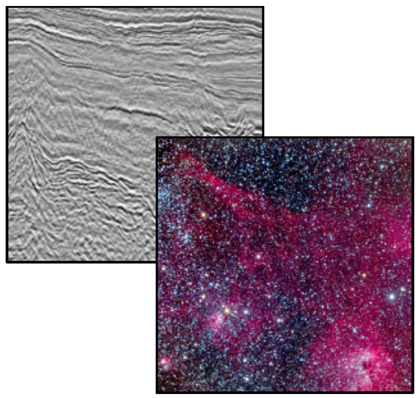

Geodesics

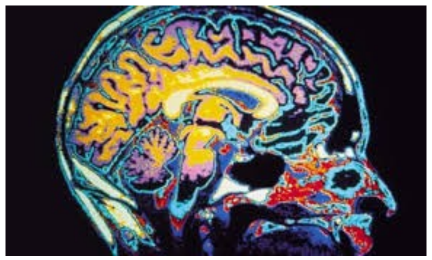

Meshes

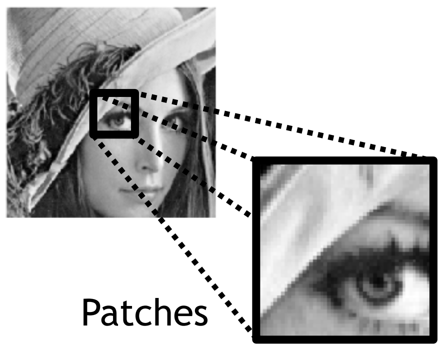

Optimization

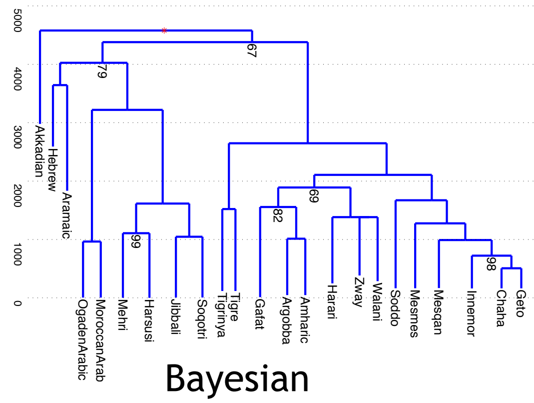


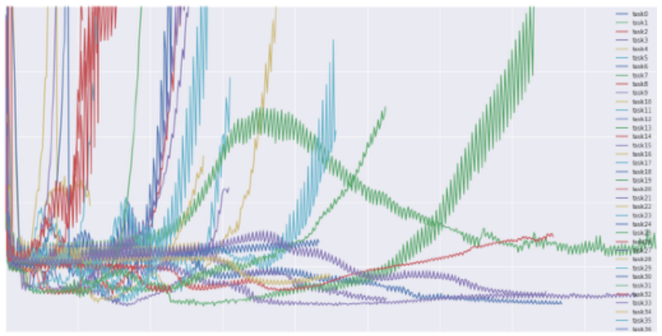
Deep Learning


Sparsity


Neuro-imaging


Patches


Bayesian


Parallel/Stochastic

Alexandre Allauzen, Paris-Sud.
Pierre Alliez, INRIA.
Guillaume Charpiat, INRIA.
Emilie Chouzenoux, Paris-Est.

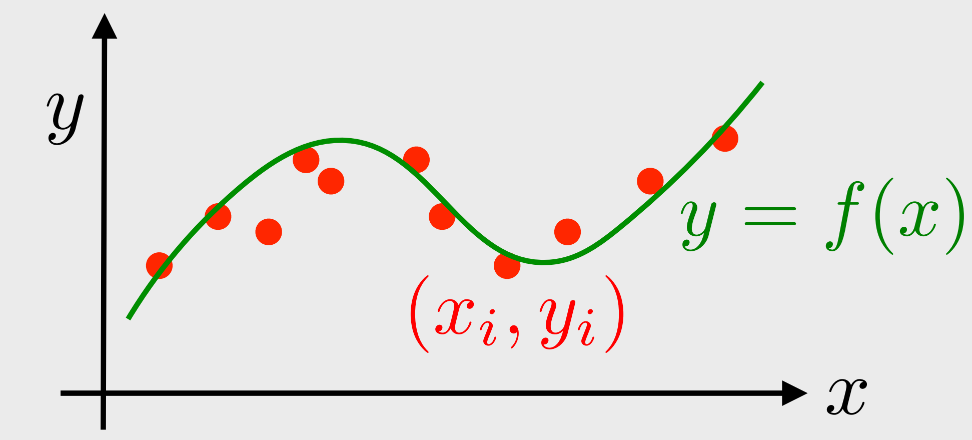
Nicolas Courty, IRISA.
Laurent Cohen, CNRS Dauphine.
Marco Cuturi, ENSAE.
Julie Delon, Paris 5.

Fabian Pedregosa, INRIA.
Julien Tierny, CNRS and P6.
Robin Ryder, Paris-Dauphine.
Gael Varoquaux, INRIA.

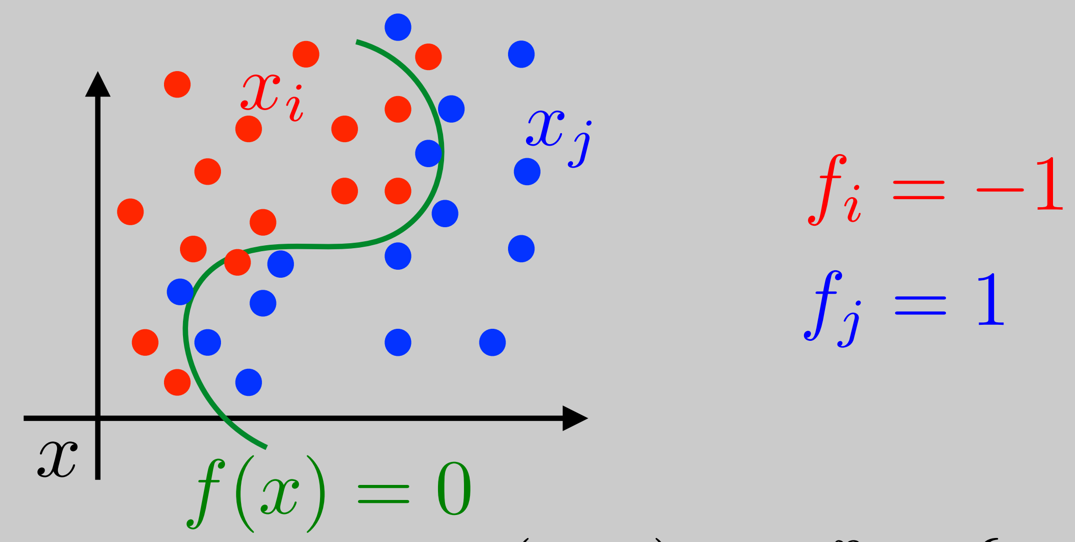
Jalal Fadili, ENSICaen.
Alexandre Gramfort, INRIA.
Matthieu Kowalski, Supelec.
Jean-Marie Mirebeau, CNRS,P-Sud.

Parametric Models

(Noisy) observations (x_i, y_j), try to infer $y = f(x)$.



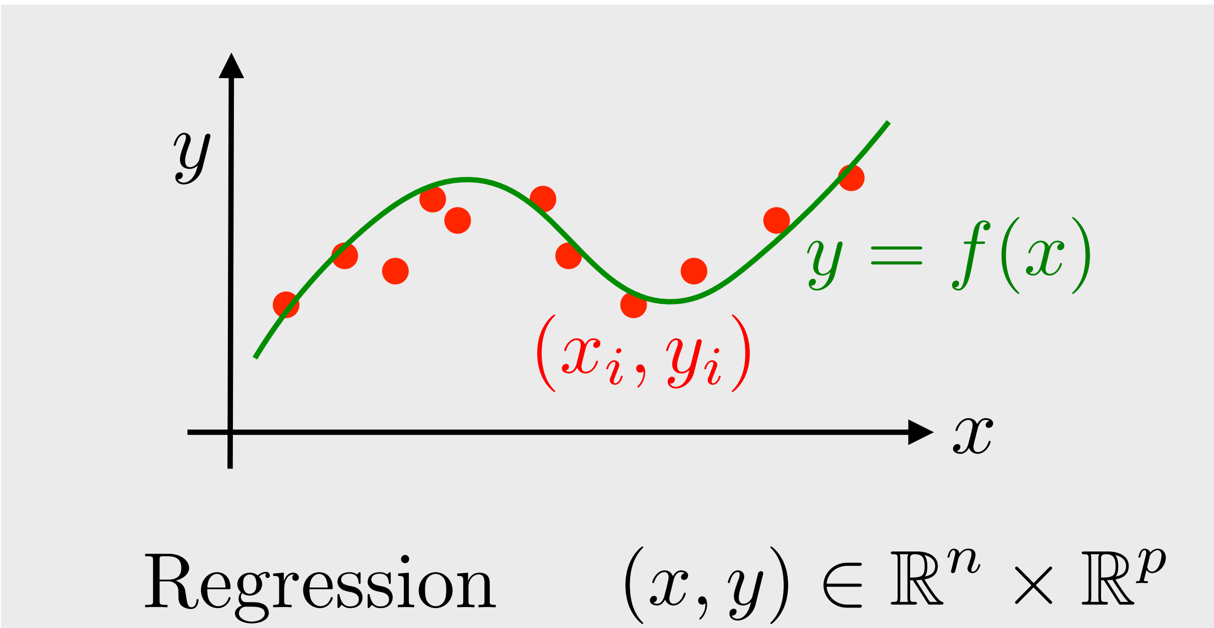
Regression $(x, y) \in \mathbb{R}^n \times \mathbb{R}^p$

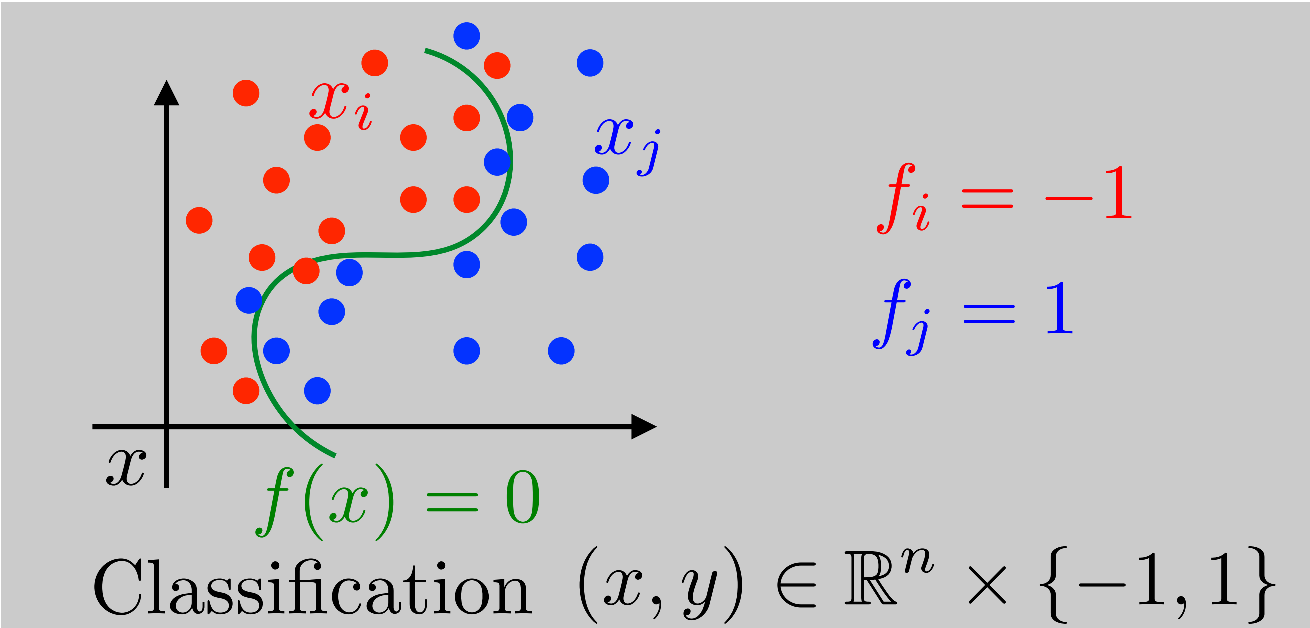


Classification $(x, y) \in \mathbb{R}^n \times \{-1, 1\}$

Parametric Models

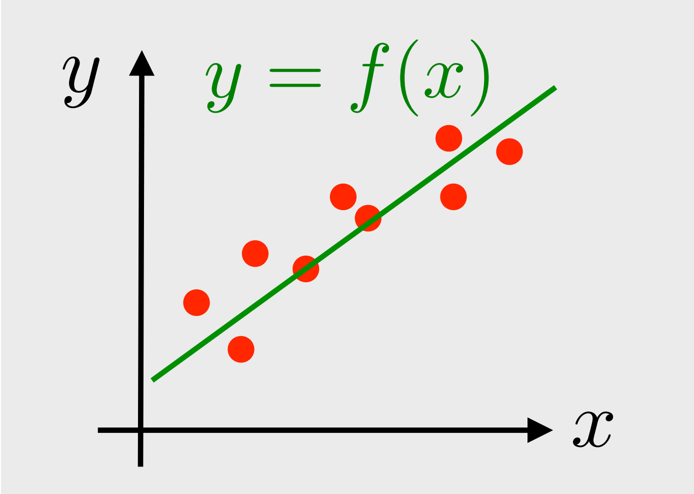
(Noisy) observations (x_i, y_j), try to infer $y = f(x)$.

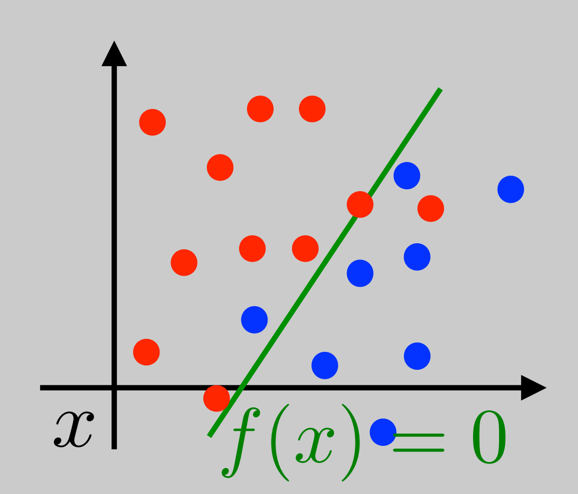




Parametric model: $y = f(x, \theta)$, find θ.

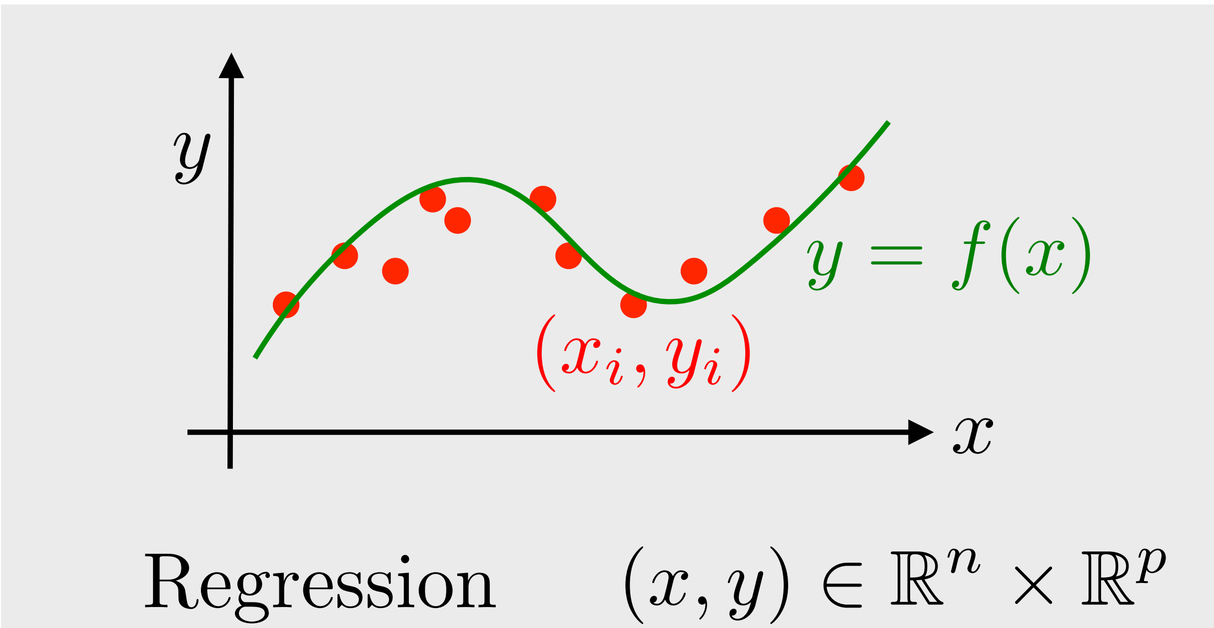
Linear model: $f(x, \theta) = \langle x, \theta \rangle$.

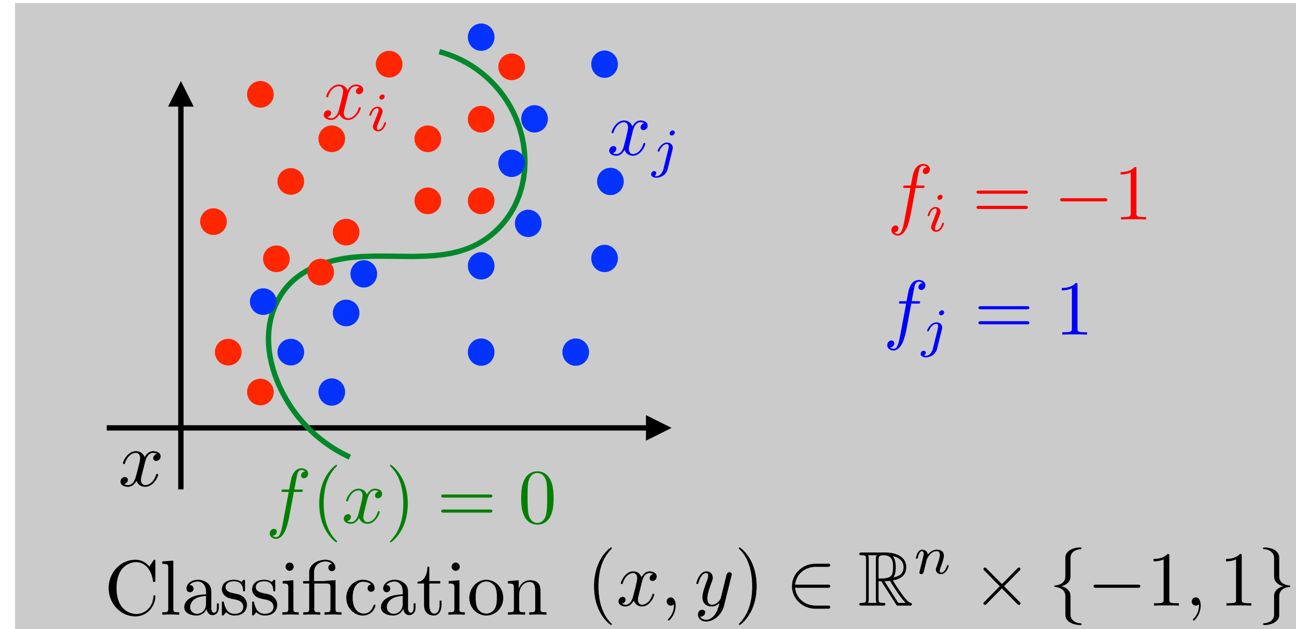




Parametric Models

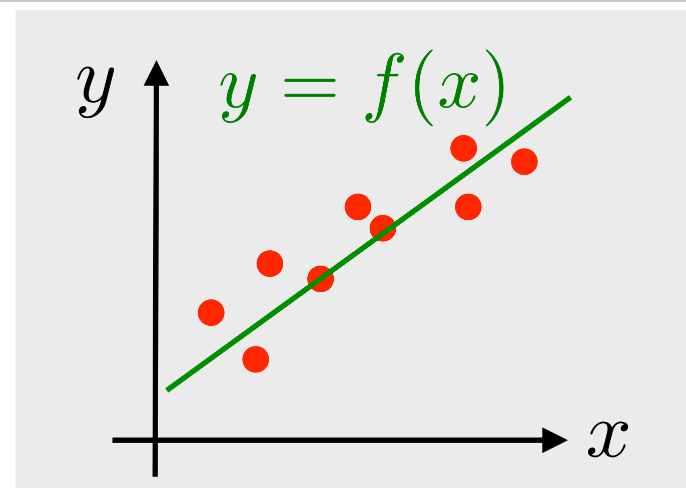
(Noisy) observations (x_i, y_j), try to infer $y = f(x)$.

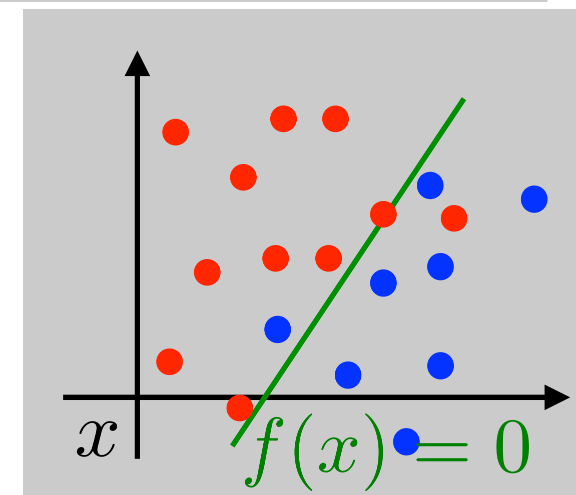




Parametric model: $y = f(x, \theta)$, find θ.

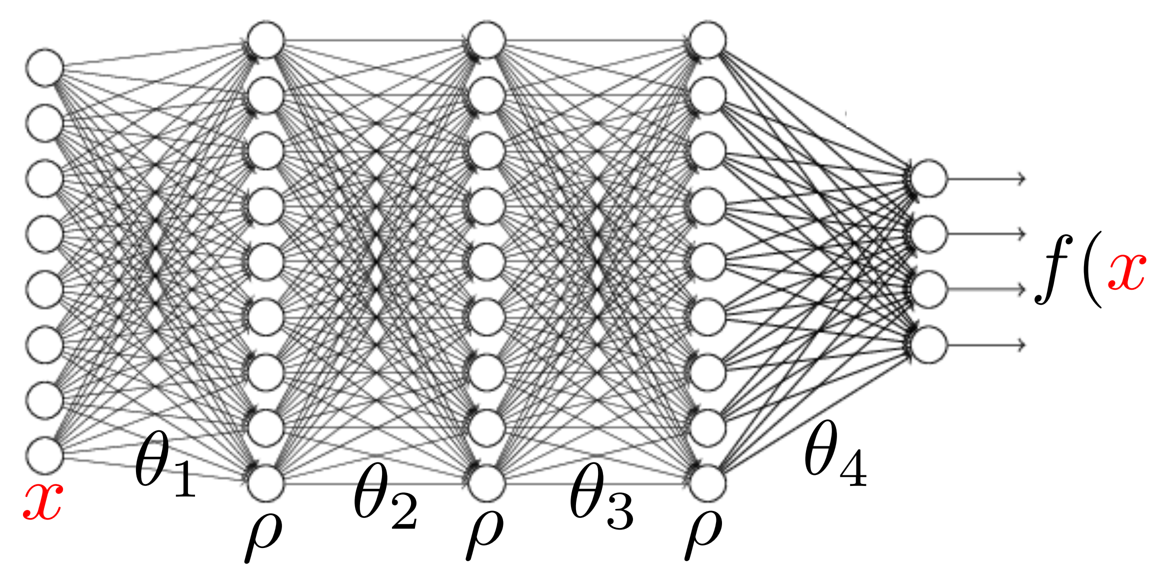
Linear model: $f(x, \theta) = \langle x,\, \theta \rangle$.





Deep network:

$$f(x, \theta) = \theta_K(\ldots \rho(\theta_2(\rho(\theta_1(x) \ldots)$$



Empirical Loss Minimization

Regression: $(y, y') \in \mathbb{R}^d \times \mathbb{R}^d$, $L(y, y') = \|y - y'\|^2$

Classification: $(y, y') \in \mathbb{R}^d \times \{-1, 1\}$, $L(y, y') = \log(\exp(-y'y) + 1)$

Loss minimization:

$$\min_\theta \sum_i L(f(x_i, \theta), y_i)$$

$$\min_\theta \mathbb{E}_{(X,Y)}(L(f(X, \theta), Y))$$

Empirical Loss Minimization

Regression: $(y, y') \in \mathbb{R}^d \times \mathbb{R}^d$, $L(y, y') = \|y - y'\|^2$

Classification: $(y, y') \in \mathbb{R}^d \times \{-1, 1\}$, $L(y, y') = \log(\exp(-y'y) + 1)$

Loss minimization:

$$\min_\theta \sum_i L(f(x_i, \theta), y_i)$$

$$\min_\theta \mathbb{E}_{(X,Y)}(L(f(X, \theta), Y))$$

Stochastic gradient descent:

– Sample: $(x, y) \in \{(x_i, y_i)\}_i$ $(x, y) \sim (X, Y)$

– Update: $\theta^{(\ell+1)} \overset{\text{def.}}{=} \theta^{(\ell)} - \tau_\ell \nabla_\theta \ell_{x,y}(\theta)$

where $\ell_{x,y}(\theta) \overset{\text{def.}}{=} L(f(x, \theta), y)$

Gradient Computation

How to compute $\nabla \ell_{x,y}(\theta)$? $\ell_{x,y}(\theta) \overset{\text{def.}}{=} L(f(x,\theta), y)$

Chain rule: $\nabla \ell_{x,y}(\theta) = [\partial f(x,\theta)]^\top (\nabla L(f(x,\theta), y))$

Linear $f(x,\theta) = \theta \times x$: $\partial f(x,\theta) = \theta$.

Non-linear $f(x,\theta)$: painful ... but $\ell_{x,y}$ it is just a computer program.

Gradient Computation

How to compute $\nabla \ell_{x,y}(\theta)$? $\qquad \ell_{x,y}(\theta) \overset{\text{def.}}{=} L(f(x,\theta), y)$

Chain rule: $\nabla \ell_{x,y}(\theta) = [\partial f(x,\theta)]^\top (\nabla L(f(x,\theta), y))$

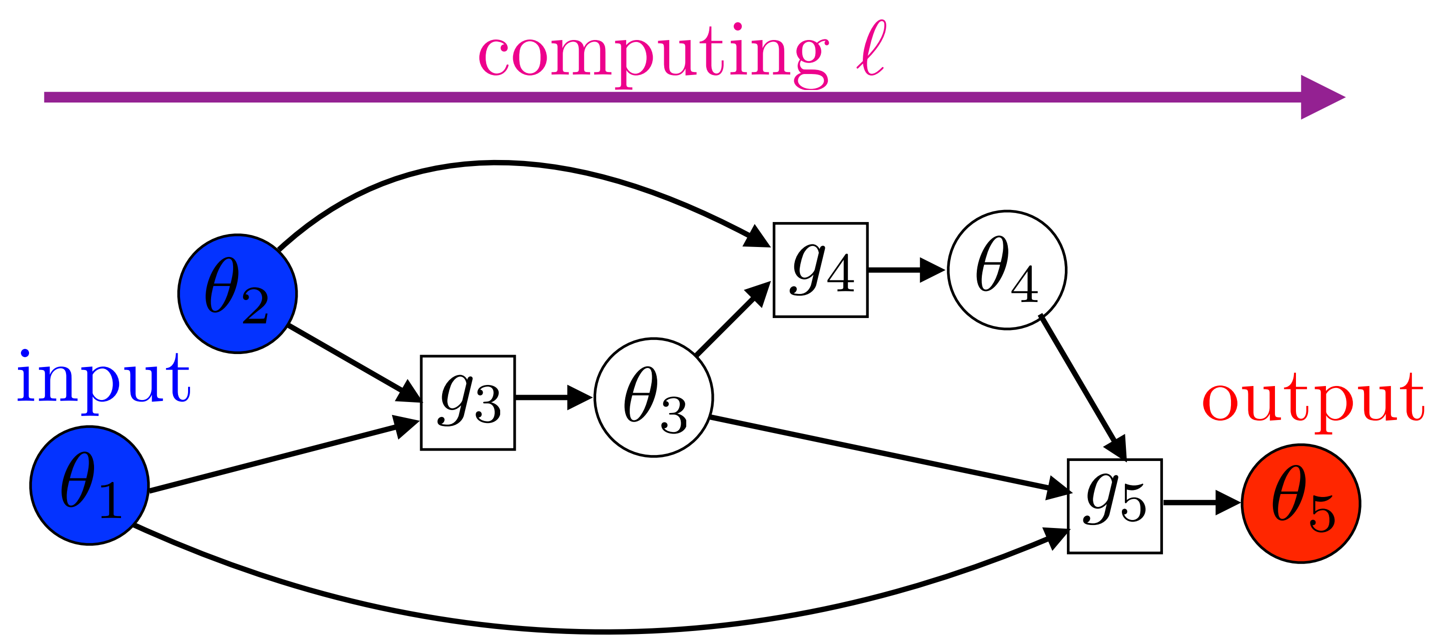
Linear $f(x,\theta) = \theta \times x$: $\partial f(x,\theta) = \theta$.

Non-linear $f(x,\theta)$: painful … but $\ell_{x,y}$ it is just a computer program.

Computer program $\Leftrightarrow$ directed acyclic graph $\Leftrightarrow$ linear ordering of nodes $(\theta_r)_r$

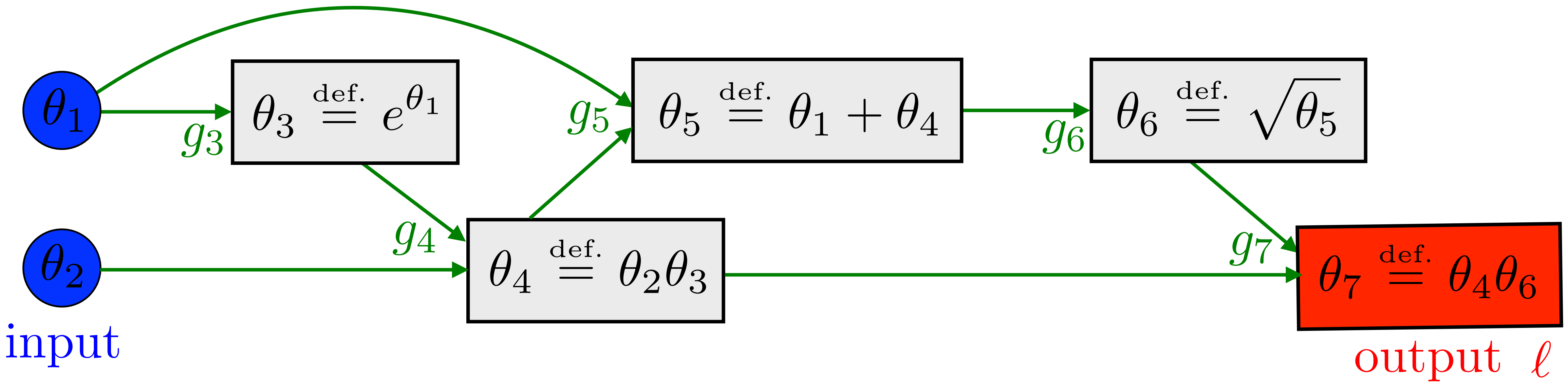
forward

function $\ell(\theta_1, \ldots, \theta_M)$
  for $r = M+1, \ldots, R$
    $\theta_r = g_r(\theta_{\text{Parents}(r)})$
  return θ_R



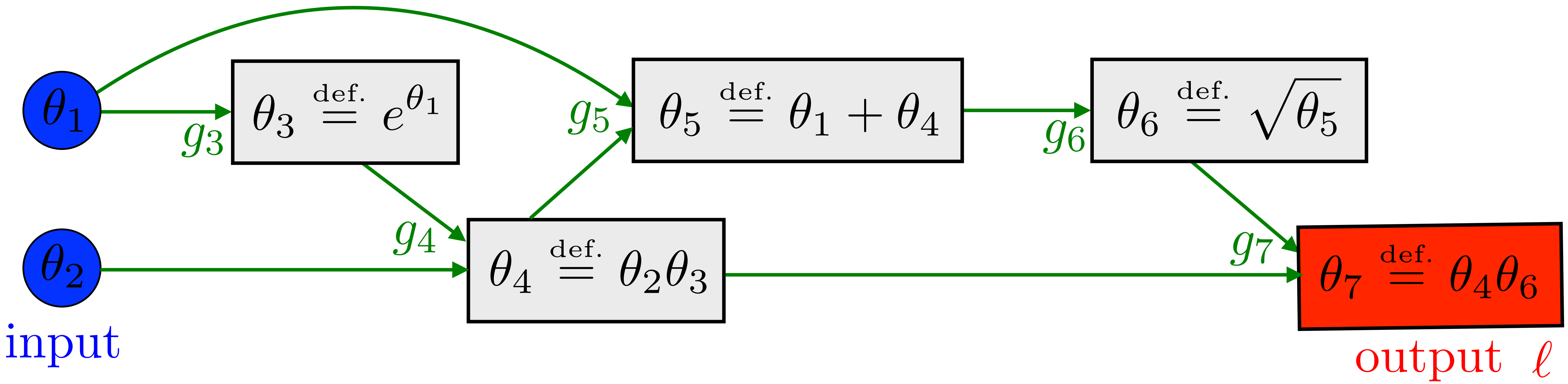
Example

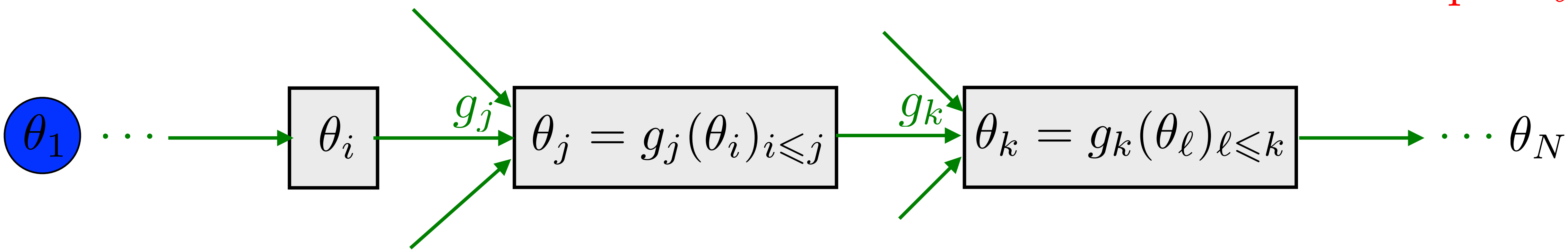
$$\ell(\theta_1, \theta_2) \stackrel{\text{def.}}{=} \theta_2 e^{\theta_1} \sqrt{\theta_1 + \theta_2 e^{\theta_1}}$$



Example

$$\ell(\theta_1, \theta_2) \overset{\text{def.}}{=} \theta_2 e^{\theta_1} \sqrt{\theta_1 + \theta_2 e^{\theta_1}}$$





Chain rules:

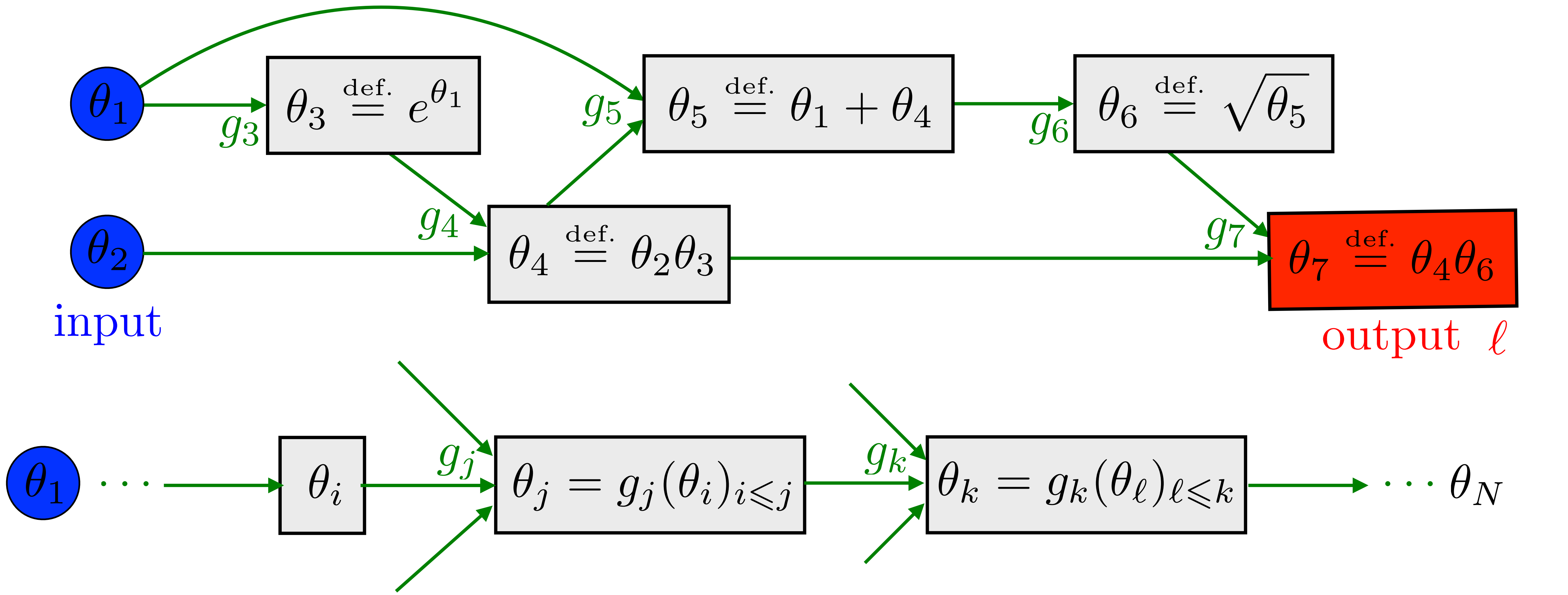
$$\text{“}\ \frac{\partial \theta_j}{\partial \theta_1} = \sum_{i \in \text{Parent}(j)} \frac{\partial \theta_j}{\partial \theta_i} \frac{\partial \theta_i}{\partial \theta_1}\ \text{”}$$

$\frac{\partial \theta_j}{\partial \theta_i} \to \partial_i g_j(\theta)$

“Classical” evaluation: **forward**.
Complexity $\sim$ #inputs.

Example

$$\ell(\theta_1, \theta_2) \overset{\text{def.}}{=} \theta_2 e^{\theta_1} \sqrt{\theta_1 + \theta_2 e^{\theta_1}}$$



Chain rules:

$$\text{“}\ \frac{\partial \theta_j}{\partial \theta_1} = \sum_{i \in \text{Parent}(j)} \frac{\partial \theta_j}{\partial \theta_i} \frac{\partial \theta_1}{\partial \theta_1}\ \text{”}$$

$\frac{\partial \theta_j}{\partial \theta_i} \rightarrow \partial_i g_j(\theta)$

“Classical” evaluation: **forward**.
Complexity $\sim$ #inputs.

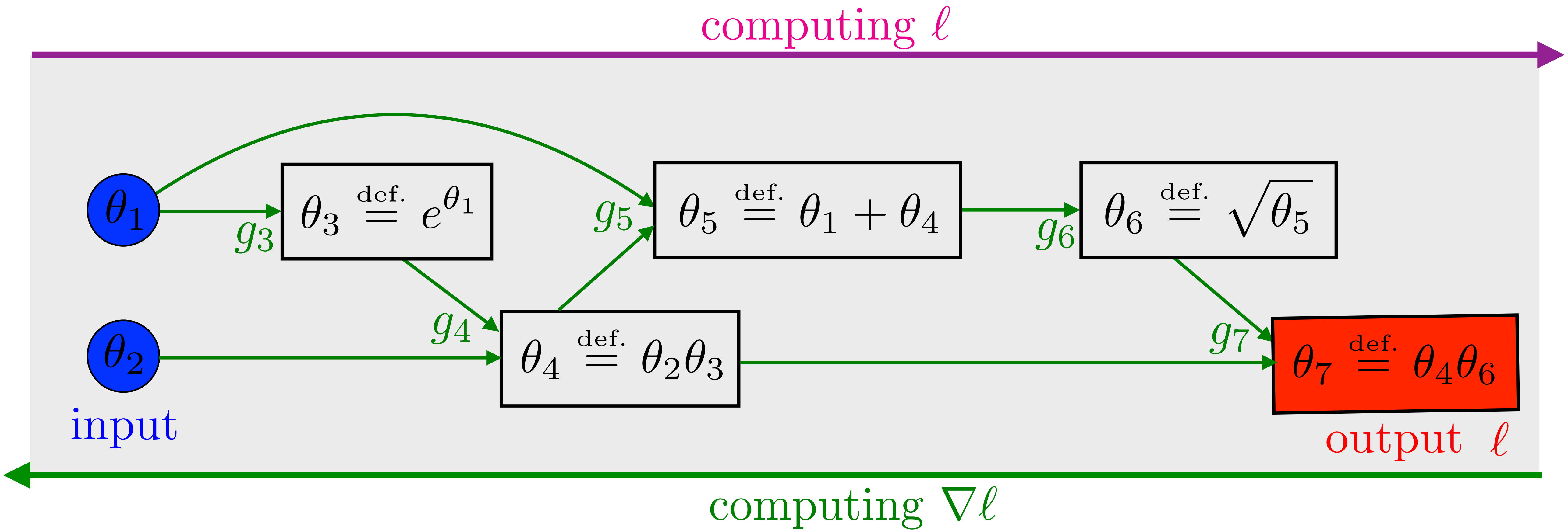
$$\text{“}\ \frac{\partial \theta_N}{\partial \theta_j} = \sum_{k \in \text{Child}(j)} \frac{\partial \theta_N}{\partial \theta_k} \frac{\partial \theta_k}{\partial \theta_j}\ \text{”}$$

$\frac{\partial \theta_N}{\partial \theta_j} \rightarrow \nabla_j \ell(\theta)$, $\frac{\partial \theta_N}{\partial \theta_k} \rightarrow \nabla_k \ell(\theta)$, $\frac{\partial \theta_k}{\partial \theta_j} \rightarrow \partial_j g_k(\theta)$

Backward evaluation.
Complexity $\sim$ #outputs (1 for grad).

Backward Automatic Differentiation

$$\ell(\theta_1, \theta_2) \overset{\text{def.}}{=} \theta_2 e^{\theta_1} \sqrt{\theta_1 + \theta_2 e^{\theta_1}}$$



forward

```
function ℓ(θ_1, ..., θ_M)
  for r = M+1, ..., R
  |   θ_r = g_r(θ_Parents(r))
  return θ_R
```

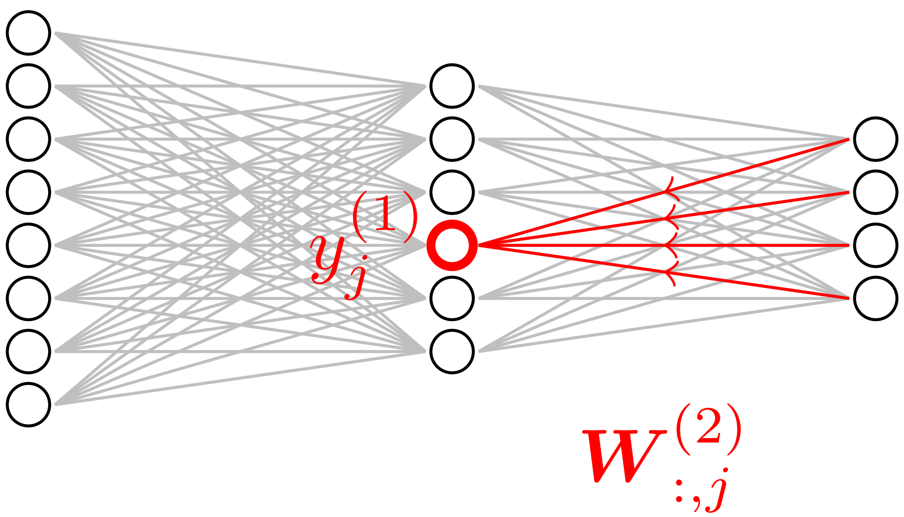
backward

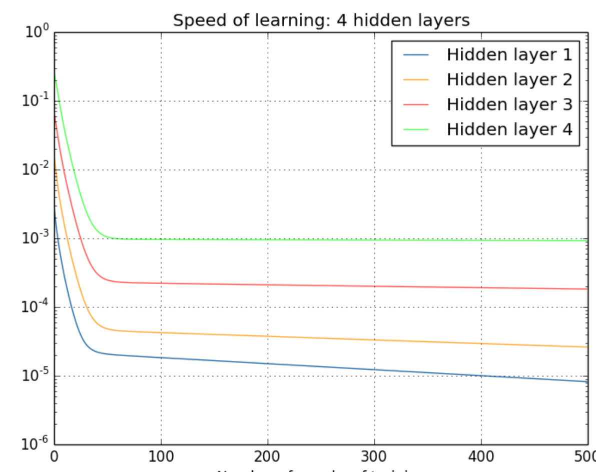
```
function ∇ℓ(θ_1, ..., θ_M)
    ∇_R ℓ = 1
    for r = R−1, ..., 1
    |   ∇_r ℓ = Σ_{s∈Child(r)} ∂_r g_s(θ) ∇_s ℓ
    return (∇_1 ℓ, ..., ∇_M ℓ)
```

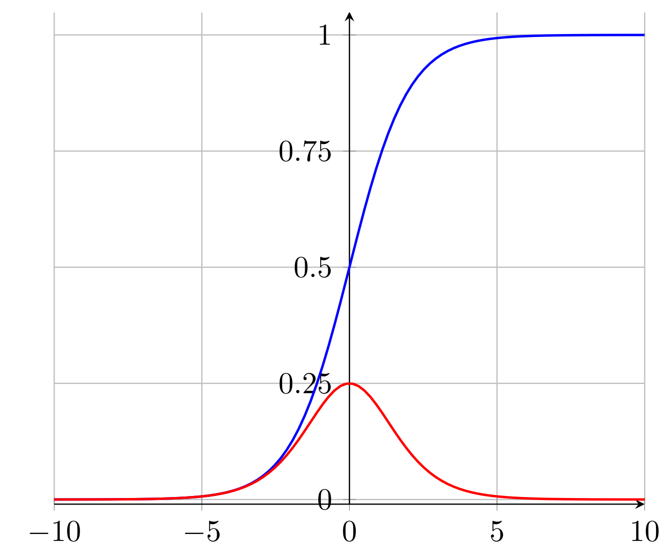
$$\nabla_r \ell = \sum_{s \in \text{Child}(r)} \partial_r g_s(\theta)\, \nabla_s \ell$$

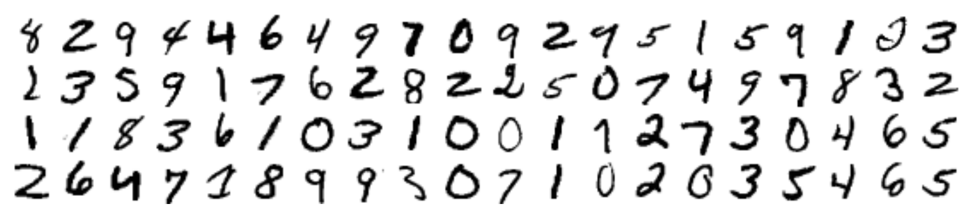
What's Next

Alexandre Allauzen: deep neural networks training.









Guillaume Charpiat: architecture of deep neural networks.