NUS DATA SCIENCE COMPETITION 2018

Modeling Weather Radar

Outline

- Radar calibration and models
- Gradient descent
- Python demo
- Competition task

Weather radar

microwave pulse

backscatter

Thunderstorm cell

Weather Radar Station

Weather radar

microwave pulse

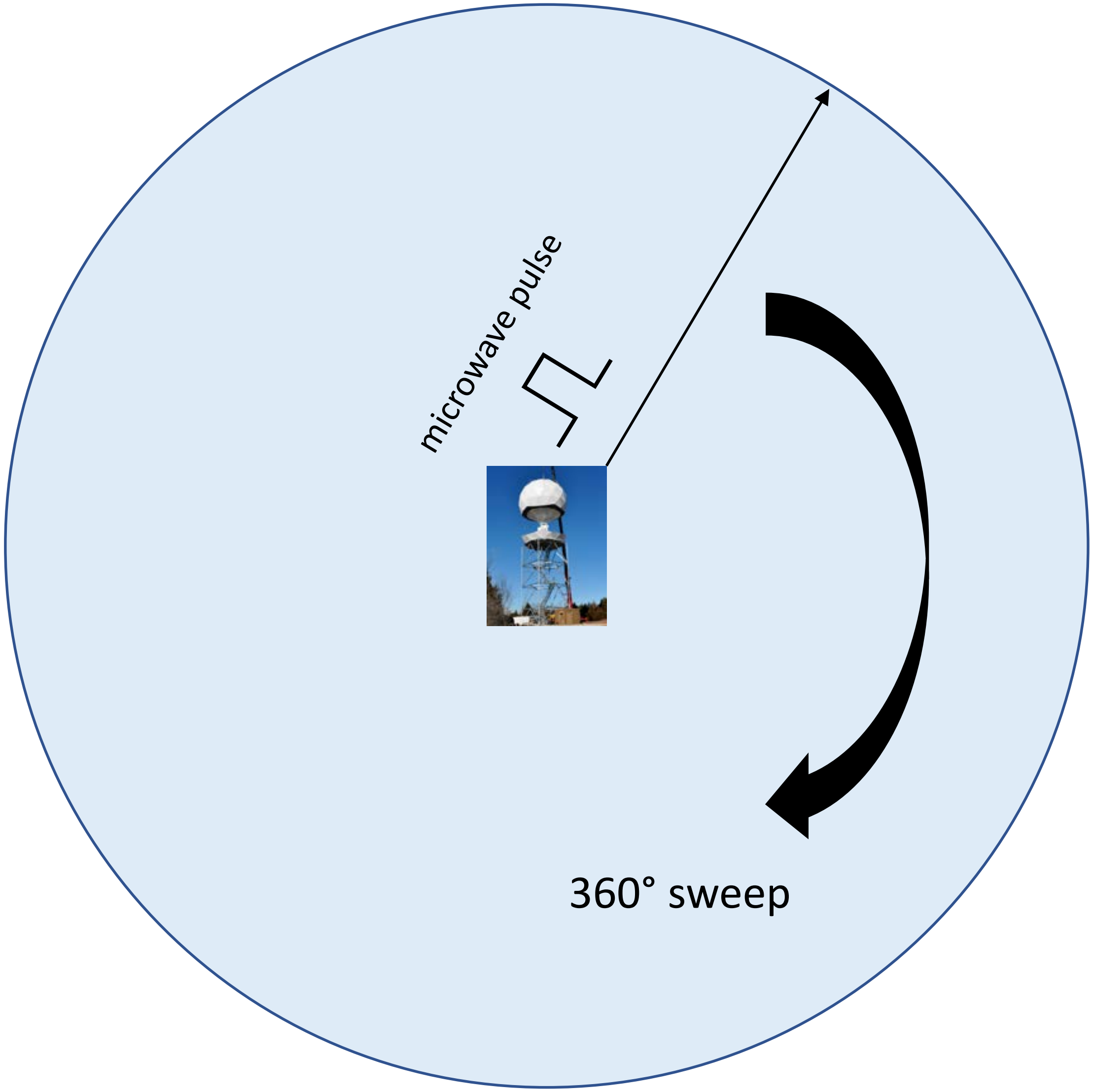
Weather Radar Station

Weather radar

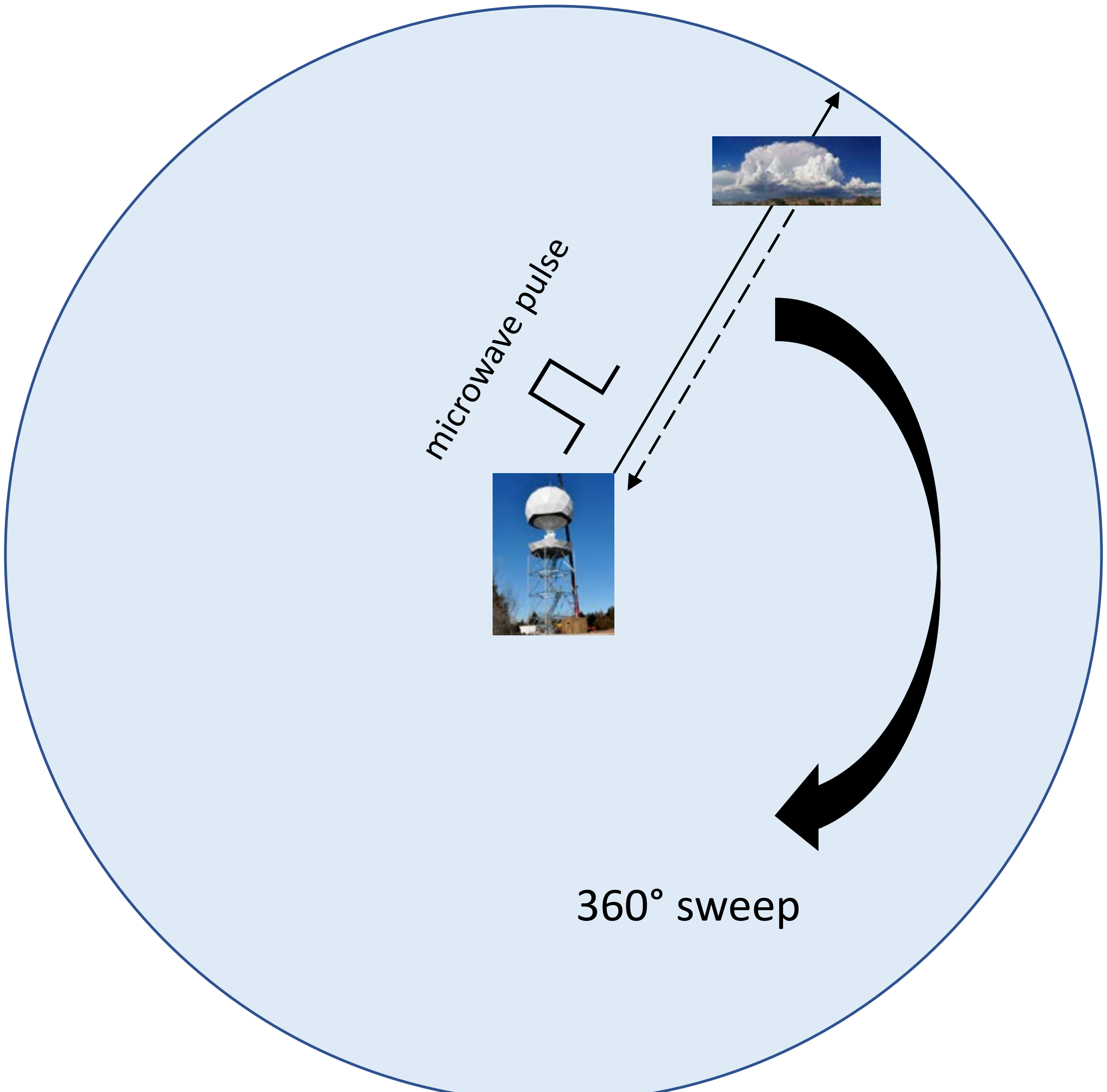
Time difference $\longrightarrow$ Distance

Intensity $\longrightarrow$ Rainfall rate

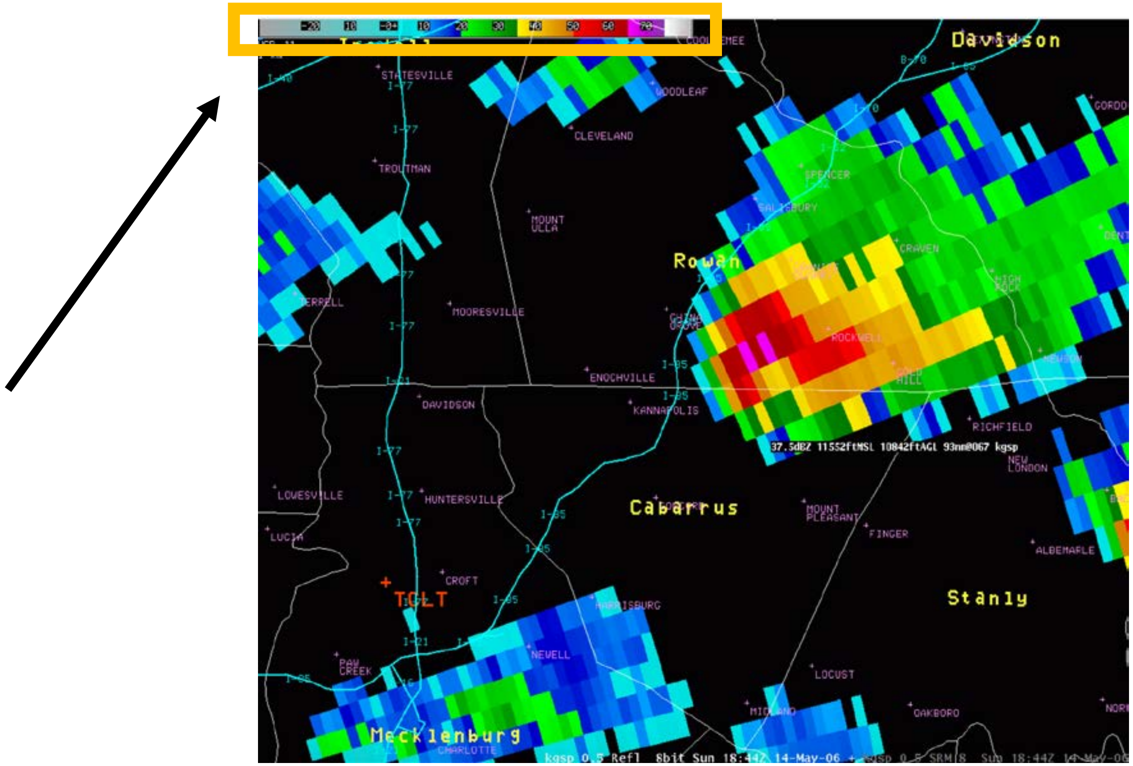
Weather radar



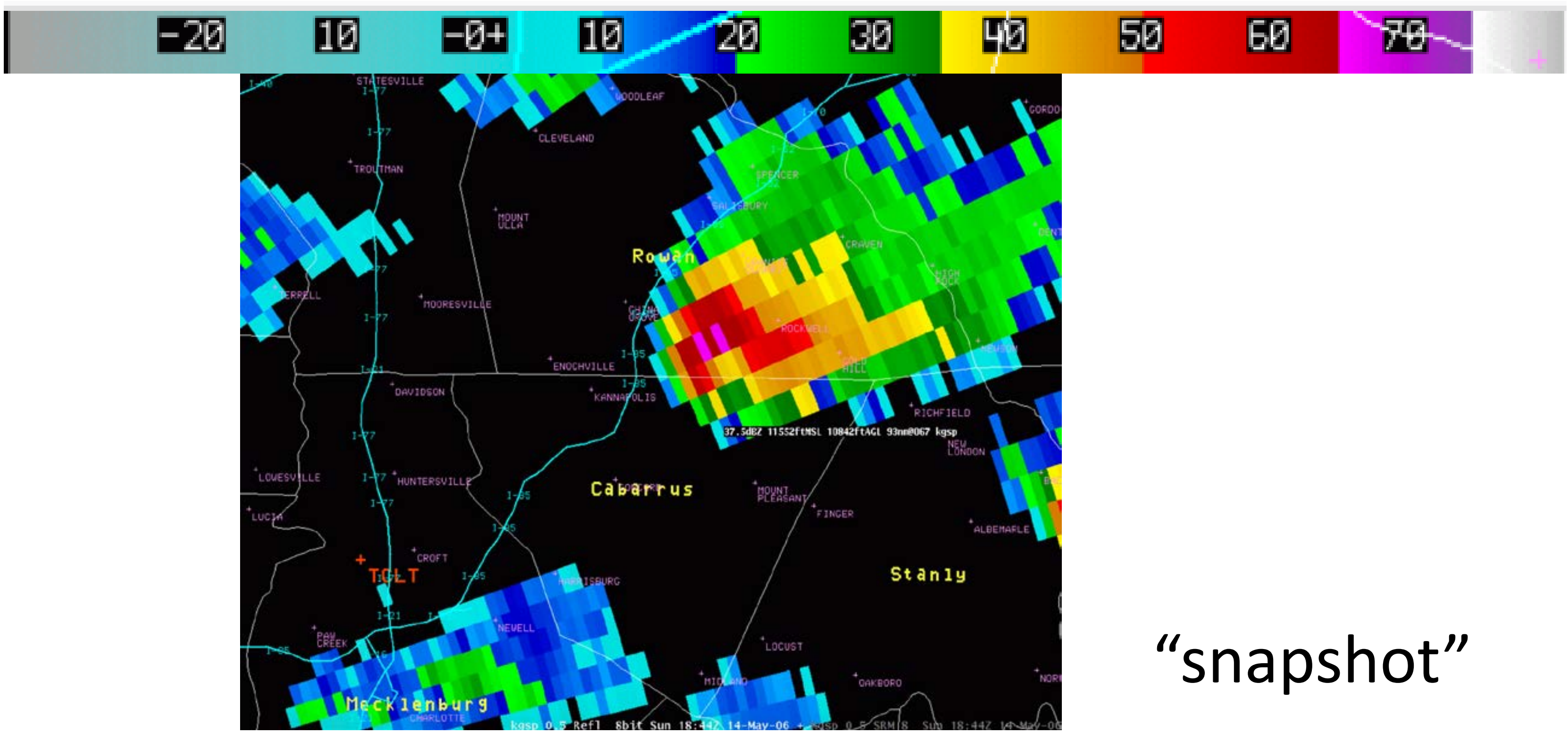
Weather radar



Weather radar



Weather radar



“snapshot”

Backscattered signal quantized into 34 bins

Rainfall model

- Convert backscatter intensity to rainfall rate at a location point in the snapshot

- A simple model: $r(x, y) = A\, e^{B \cdot k(x,y)}$

x, y	: location coordinates
$r(x, y)$	: rainfall rate at location (x, y)
$k(x, y)$	: bin number of radar signal at (x, y)
A, B	: model parameters

Rainfall model

- Convert to weekly cumulative rainfall R_{radar}
- Radar snapshots taken at a regular time interval Δt

$$R_{radar}(x,y) = \sum_k r(x,y)c_k(x,y)\,\Delta t$$

$$\Big\downarrow\; r(x,y) = A\,e^{B\cdot k(x,y)}$$

$$= \sum_k A\,e^{B\cdot k(x,y)} c_k(x,y)\,\Delta t$$

c_k : weekly bin counts

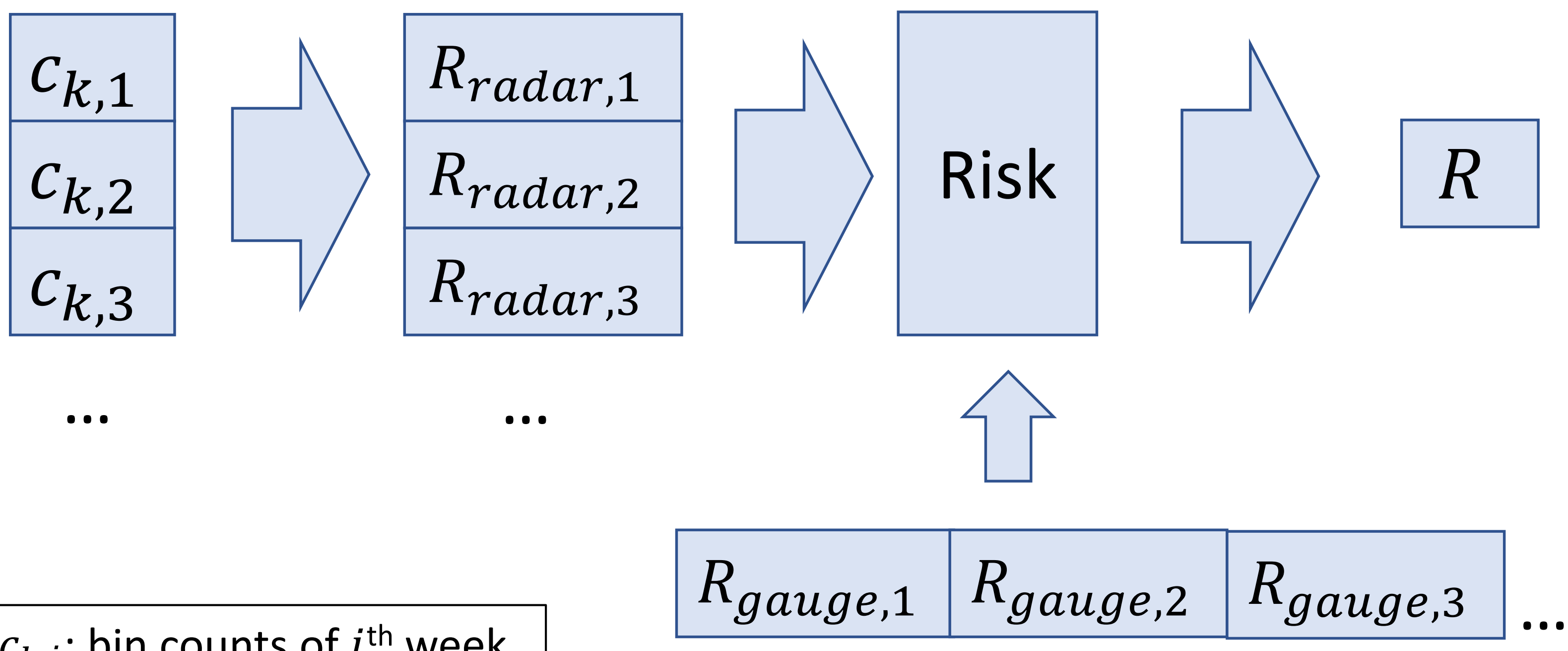
Rainfall model

- Calibrate rainfall model with rain gauge data

- Rain gauge
  - Rainfall measuring devices
  - Located at fixed facilities
  - Daily readings accumulated by week

Risk

- Error measure between rain gauge readings and rainfall deduced from radar



$c_{k,i}$: bin counts of i^{th} week

Risk

$$L_i = \left(R_{radar,i} - R_{gauge,i}\right)^2$$

$$R = \frac{1}{T}\sum_i L_i$$

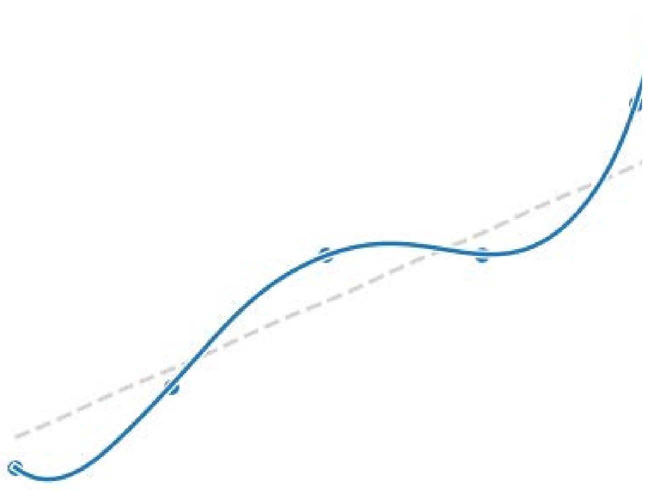
Risk minimization

1. Initialize A and B to some values (eg. some random number)
2. Calculate $R_{radar,w}$ for all weeks w in total time period T
3. Calculate the risk R and save it as R_0
4. *Somehow adjust* A and B so that the new risk $R_1 < R_0$. If this can't be done, stop. If yes, save R_1 as R_0 and repeat this step.

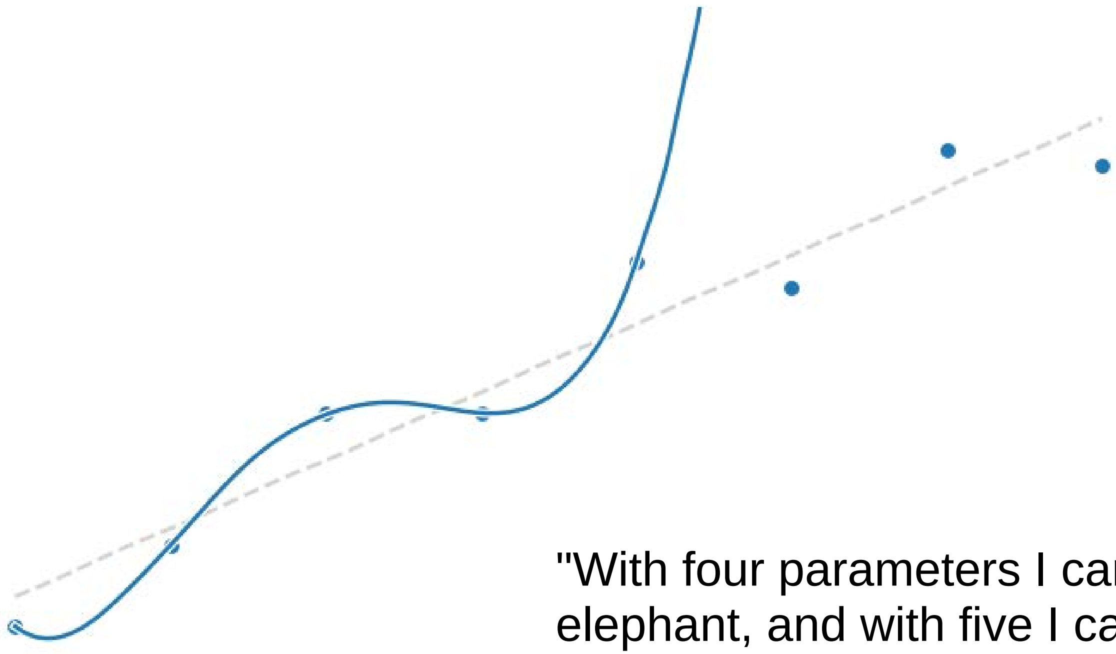
Risk minimization

- Possible outcomes
  - High final risk
    - Problem with model
    - Problem with learning algorithm (eg. local minima)
  - Low/zero final risk
    - Possible overfitting

Overfit



Overfit

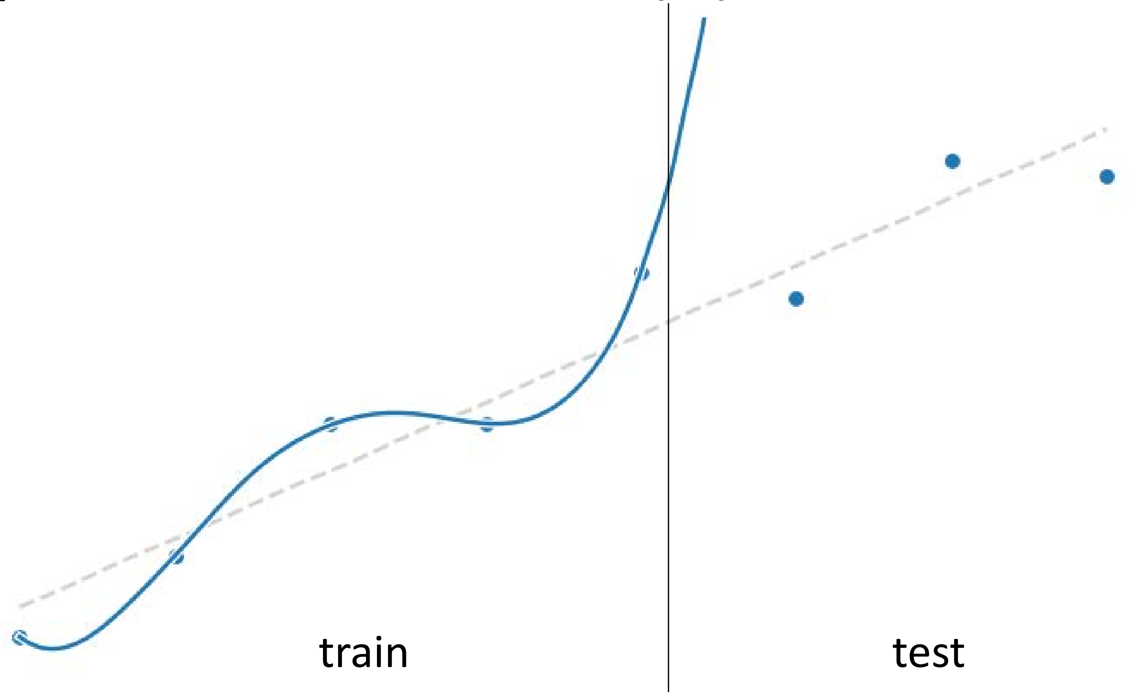


"With four parameters I can fit an elephant, and with five I can make him wiggle his trunk"

John von Neumann

Overfit

- Split dataset into train (D) and test sets



Risk minimization: evaluate risk over the train set

Report: use risk over test set as performance measure

Risk minimization

1. Initialize A and B to some values (eg. some random number)
2. Calculate $R_{radar,w}$ for all weeks w in total time period T
3. Calculate the risk R and save it as R_0
4. *Somehow adjust* A and B so that the new risk $R_1 < R_0$. If this can't be done, stop. If yes, save R_1 as R_0 and repeat this step.

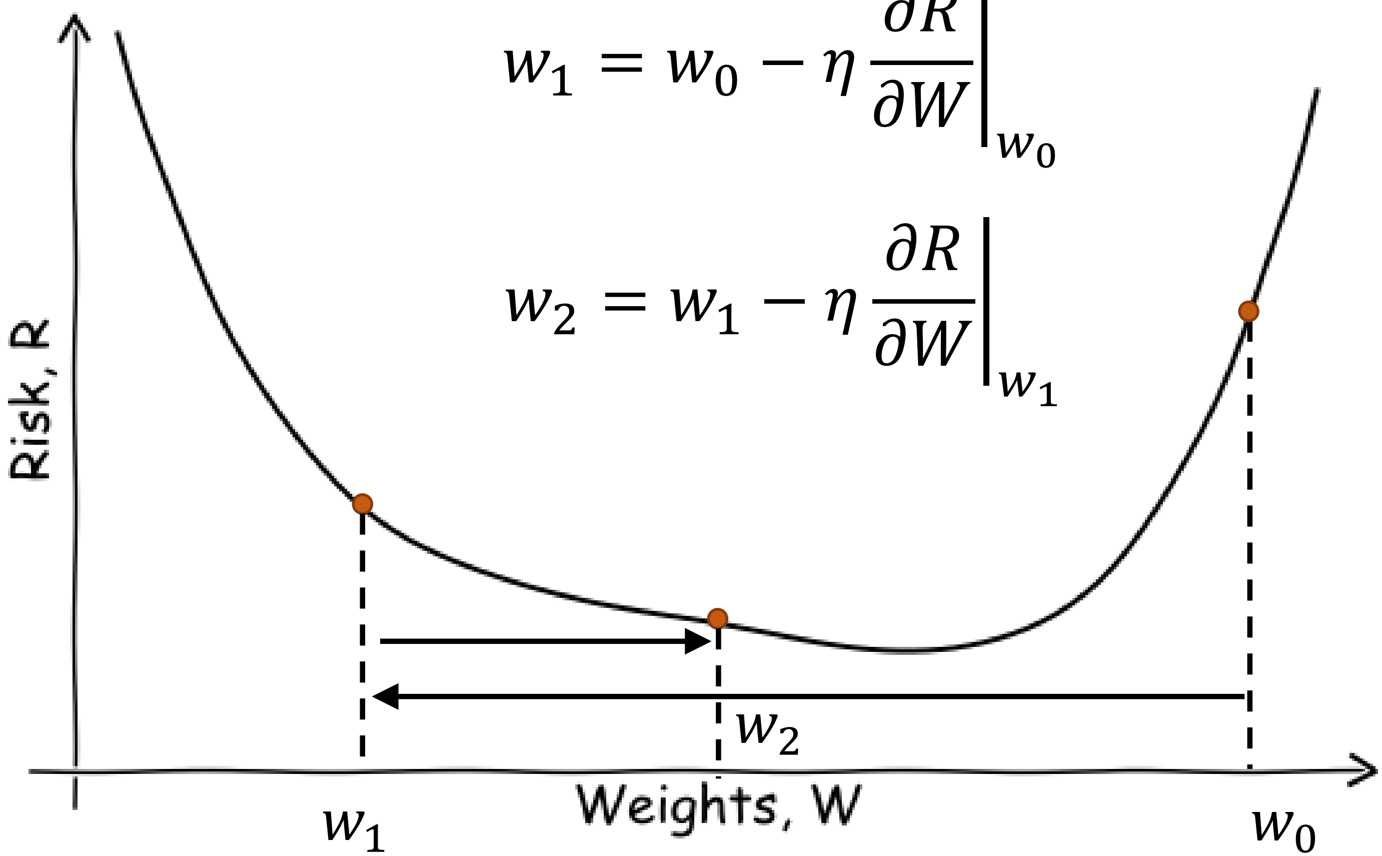
Gradient descent

- Move in direction of steepest downward slope $(-\frac{\partial R}{\partial W})$
- Weights = model parameters (ie. A and B)

$$W_{new} = W_{old} - \eta \left.\frac{\partial R}{\partial W}\right|_{W_{old}}$$

Learning rate

Gradient descent



$$w_1 = w_0 - \eta \left. \frac{\partial R}{\partial W} \right|_{w_0}$$

$$w_2 = w_1 - \eta \left. \frac{\partial R}{\partial W} \right|_{w_1}$$

Gradient descent

$$w_1 = w_0 - \eta \left.\frac{\partial R}{\partial W}\right|_{w_0}$$

$$\boxed{\delta W = -\eta \frac{\partial R}{\partial W}}$$

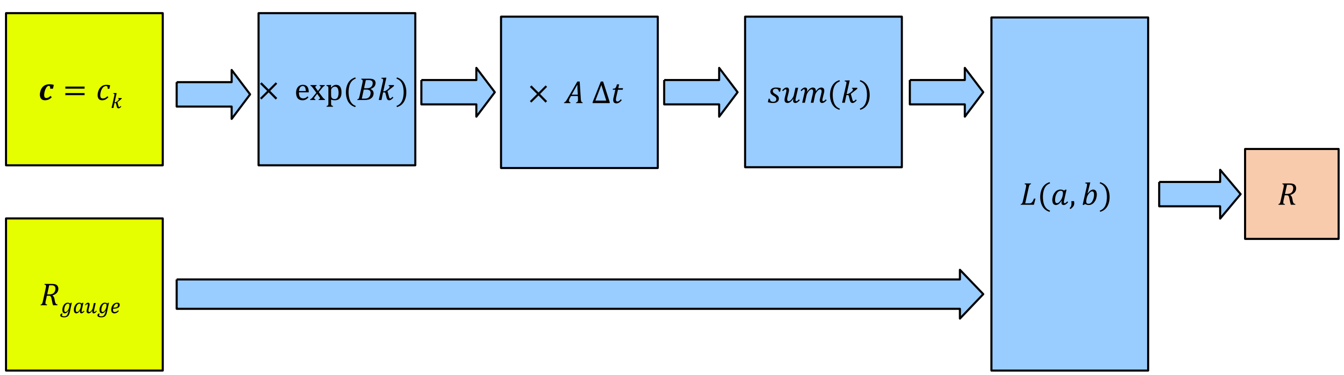
$$\begin{aligned} R_{i+1} &= R(A + \delta A, B + \delta B) \\ &\approx R(A, B) + \delta A \frac{\partial R}{\partial A} + \delta B \frac{\partial R}{\partial B} \\ &= R(A, B) - \eta \left(\frac{\partial R}{\partial A}\right)^2 - \eta \left(\frac{\partial R}{\partial B}\right)^2 \\ &\geq R(A, B) = R_i \end{aligned}$$

Backpropagation

- Goal: find $\frac{\partial R}{\partial W}$

$$R = \frac{1}{T}\sum_i \left(R_{radar,i} - R_{gauge,i}\right)^2$$

$$R_{radar}(x,y) = \sum_k A\, e^{B\cdot k(x,y)} c_k(x,y)\,\Delta t$$



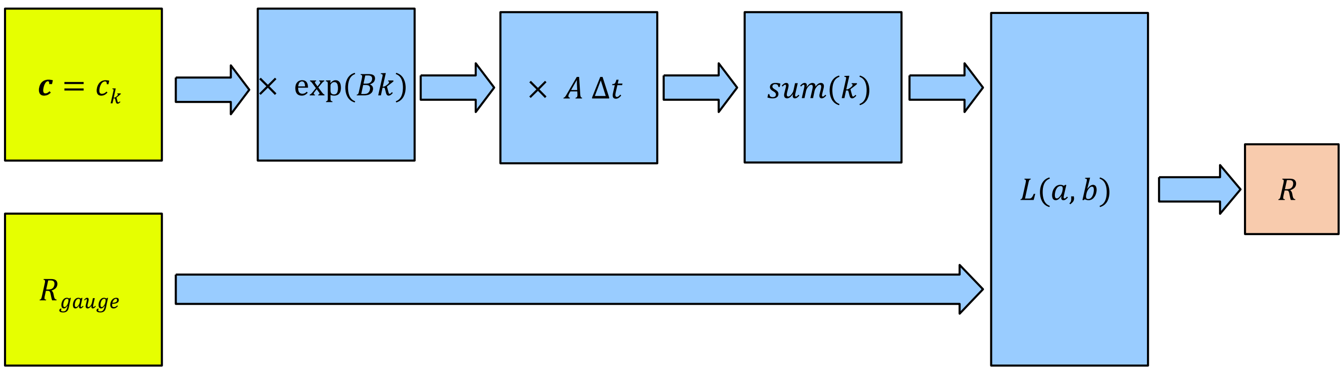
Backpropagation

$$R = R\big(L(a,b)\big)$$

$$\frac{\partial R}{\partial W} = \frac{\partial R}{\partial L}\frac{\partial L}{\partial W}$$

$$= \frac{1}{T}\sum_i \frac{\partial L_i}{\partial W}$$

$$\boxed{R = \frac{1}{T}\sum_i L_i}$$



Backpropagation

Consider a series of operations $\{s_1, s_2, \dots, s_N\}$, each operation having a parameter W_N and output X_N

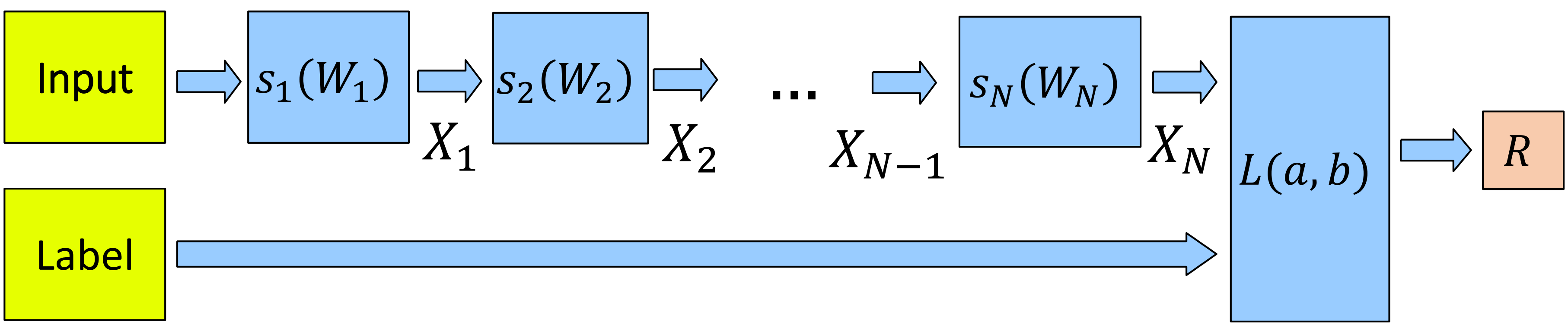
$$\frac{\partial R}{\partial W_1} = \frac{1}{T}\sum_i \frac{\partial L_i}{\partial W_1}$$

$$\frac{\partial L_i(W_N, X_N)}{\partial W_1} = \frac{\partial L}{\partial X_N}\frac{\partial X_N}{\partial W_1}$$

$$= \frac{\partial L}{\partial X_N}\frac{\partial X_N}{\partial X_{N-1}}\frac{\partial X_{N-1}}{\partial W_1}$$

$$\boxed{X_N(X_{N-1}, W_N)}$$

$$\boxed{X_{N-1}(X_{N-2}, W_{N-1})}$$

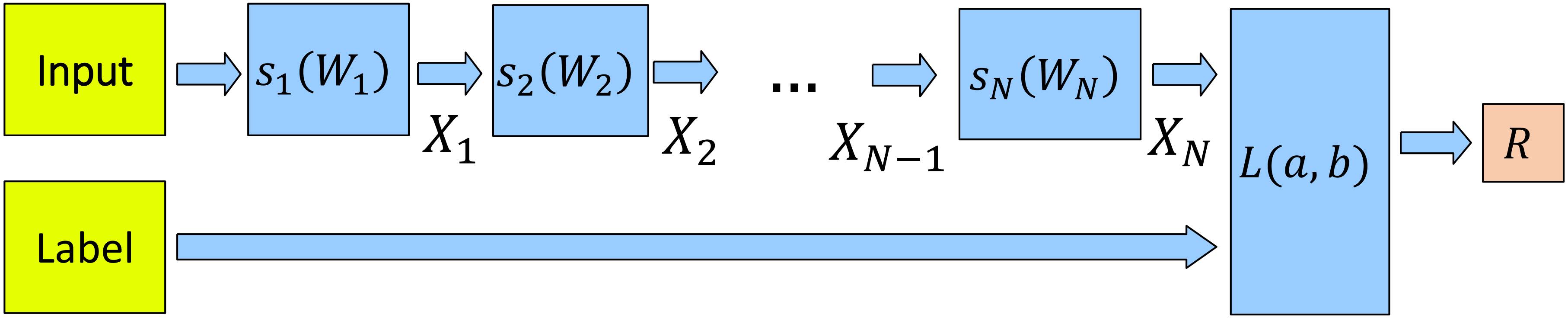


Backpropagation

Consider a series of operations $\{s_1, s_2, \dots, s_N\}$, each operation having a parameter W_N and output X_N

$$\frac{\partial R}{\partial W_1} = \frac{1}{T}\sum_i \frac{\partial L_i}{\partial W_1} \qquad \frac{\partial L_i(W_N, X_N)}{\partial W_1} = \frac{\partial L}{\partial X_N}\frac{\partial X_N}{\partial W_1}$$

$$= \frac{\partial L}{\partial X_N}\frac{\partial X_N}{\partial X_{N-1}}\frac{\partial X_{N-1}}{\partial X_{N-2}}\frac{\partial X_{N-2}}{\partial W_1}$$

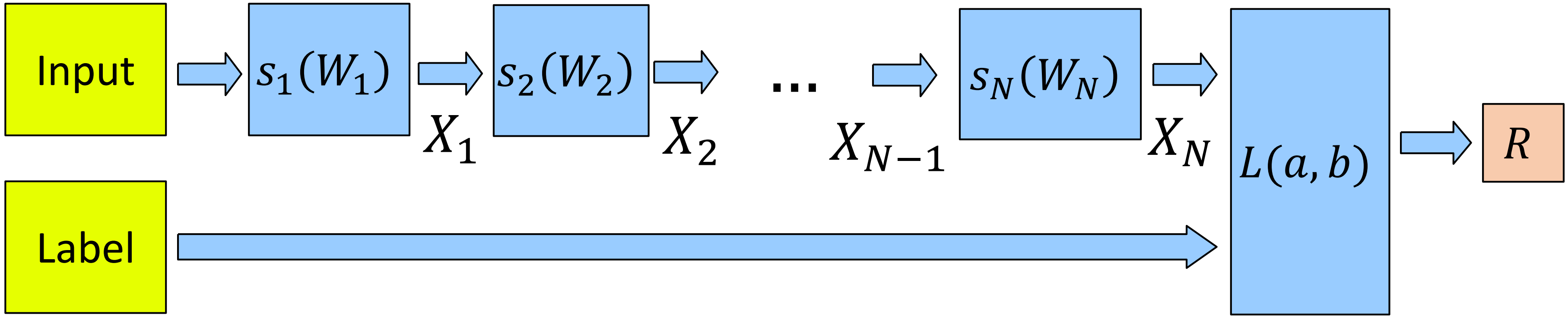


Backpropagation

Consider a series of operations $\{s_1, s_2, \dots, s_N\}$, each operation having a parameter W_N and output X_N

$$\frac{\partial R}{\partial W_1} = \frac{1}{T}\sum_i \frac{\partial L_i}{\partial W_1} \qquad \frac{\partial L_i(W_N, X_N)}{\partial W_1} = \frac{\partial L}{\partial X_N}\frac{\partial X_N}{\partial W_1}$$

$$= \frac{\partial L}{\partial X_N}\frac{\partial X_N}{\partial X_{N-1}}\frac{\partial X_{N-1}}{\partial X_{N-2}} \cdots \frac{\partial X_2}{\partial X_1}\frac{\partial X_1}{\partial W_1}$$



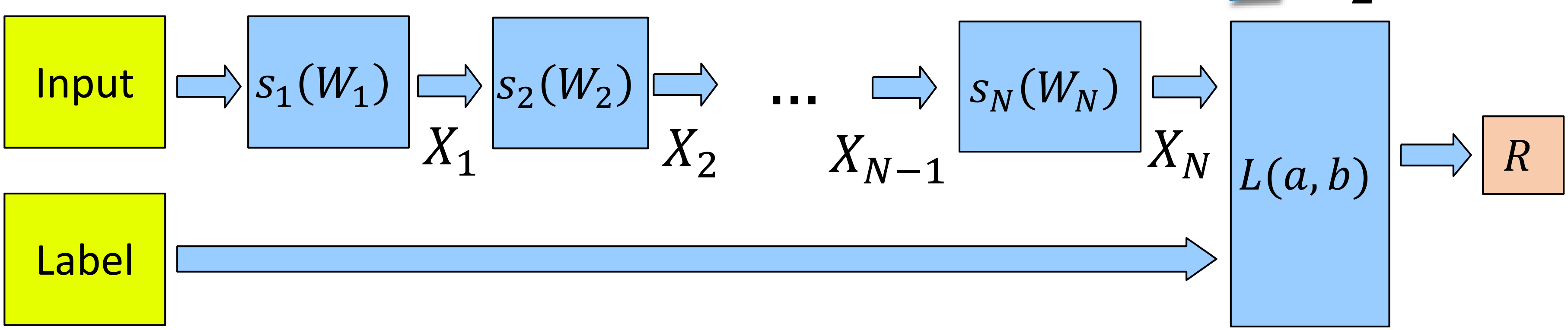
Backpropagation

Consider a series of operations $\{s_1, s_2, \dots, s_N\}$, each operation having a parameter W_N and output X_N

$$\frac{\partial R}{\partial W_1} = \frac{1}{T}\sum_i \frac{\partial L_i}{\partial W_1}$$

$$\frac{\partial L_i(W_N, X_N)}{\partial W_1} = \frac{\partial L}{\partial X_N}\frac{\partial X_N}{\partial W_1} = \underbrace{\frac{\partial L}{\partial X_N}}_{\epsilon}\underbrace{\frac{\partial X_N}{\partial X_{N-1}}}_{\delta_N}\underbrace{\frac{\partial X_{N-1}}{\partial X_{N-2}}}_{\delta_{N-1}}\cdots\underbrace{\frac{\partial X_2}{\partial X_1}}_{\delta_2}\frac{\partial X_1}{\partial W_1}$$

$$\boxed{\delta_i = \frac{\partial X_i}{\partial X_{i-1}}}$$

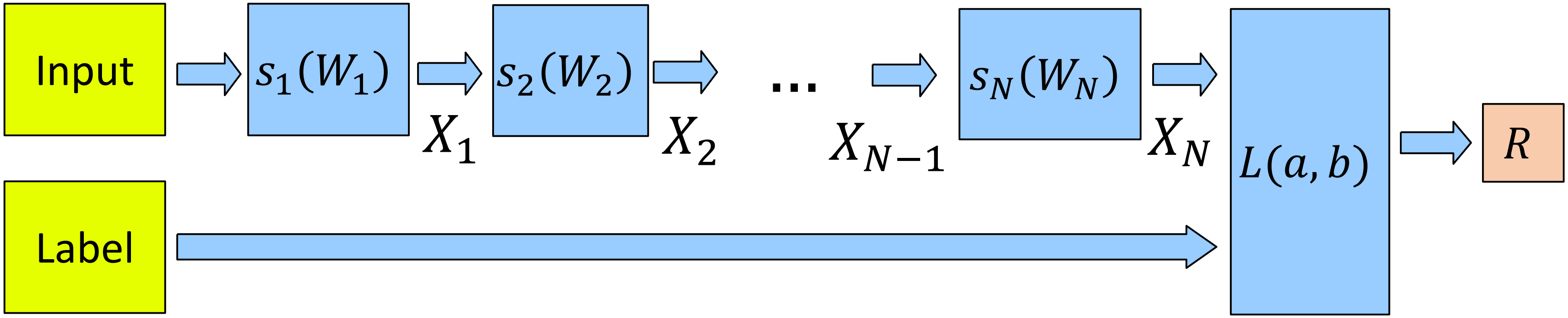


Backpropagation

Consider a series of operations $\{s_1, s_2, \dots, s_N\}$, each operation having a parameter W_N and output X_N

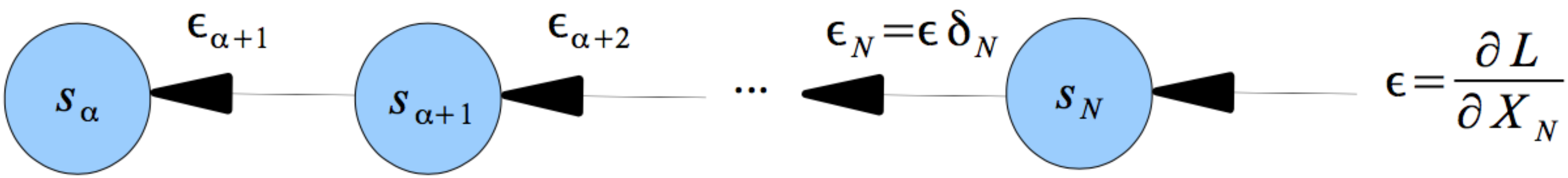
$$\frac{\partial R}{\partial W_1} = \frac{1}{T}\sum_i \frac{\partial L_i}{\partial W_1} \qquad \frac{\partial L_i(W_N, X_N)}{\partial W_1} = \epsilon\, \delta_N\, \delta_{N-1} \dots \delta_2 \frac{\partial X_1}{\partial W_1}$$

$$\boxed{\begin{aligned} \epsilon &= \frac{\partial L}{\partial X_N} \\ \delta_i &= \frac{\partial X_i}{\partial X_{i-1}} \end{aligned}}$$



Backpropagation

Backward propagation of “errors”



1 iteration = forward pass + backward pass

Backpropagation

Consider a series of operations $\{s_1, s_2, \dots, s_N\}$, each operation having a parameter W_N and output X_N

$$\frac{\partial R}{\partial W_j} = E_D\left[\epsilon_{j+1}\frac{\partial X_1}{\partial W_j}\right]$$

$$\epsilon_{j+1} = \epsilon\ \delta_N\ \delta_{N-1} \dots \delta_{j+1}$$

$$\delta_i = \frac{\partial X_i}{\partial X_{i-1}}$$

$$\epsilon = \frac{\partial L}{\partial X_N}$$

Backpropagation

Quiz: find $\frac{\partial R}{\partial A}$ and $\frac{\partial R}{\partial B}$

$$R = \frac{1}{T}\sum_i \left(R_{radar,i} - R_{gauge,i}\right)^2$$

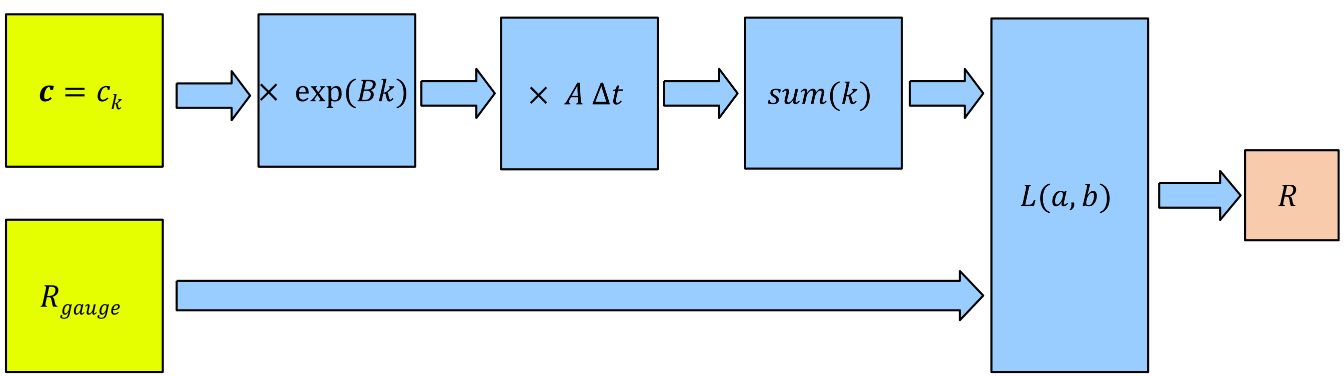
$$R_{radar}(x,y) = \sum_k A\, e^{B\cdot k(x,y)} c_k(x,y)\, \Delta t$$

$$\frac{\partial R}{\partial W_j} = E_D\left[\epsilon_{j+1}\frac{\partial X_1}{\partial W_j}\right]$$

$$\epsilon_{j+1} = \epsilon\, \delta_N\, \delta_{N-1} \ldots \delta_{j+1}$$

$$\delta_i = \frac{\partial X_i}{\partial X_{i-1}}$$

$$\epsilon = \frac{\partial L}{\partial X_N}$$



Python Demo

- Gradient descent
  - Quick and simple implementation
  - Two classes: Operation and Path
- Operation
  - Represents a node in the backpropagation network
  - Handles forward and backward passes within the operation
  - May contain a weight – one only
- Path
  - Links all the operations together
  - Includes Euclidean loss
  - Passes operation outputs and errors between operations

Python Demo

- Operation

  Without weights:

  ```
  Operation(forward, backward)
  ```

  With weights:

  ```
  Operation(forward, backward,
  learning_rate, [w_initial], [dx/dw])
  ```

- Path

  ```
  Path([operations], input, label)
  ```

Python Demo

- Example: $y=Ae^{Bx}$

op1: $y=Bx$	$y'=B$	$Y'= x$
op2: $y=e^x$	$y'= e^x$	
op3: $y=Ax$	$y'=A$	$Y'= x$

```
op1 = Operation(lambda x,w: x*w,     lambda x,w: w,
                 0.01, [0.2], [lambda x,w: x])

op2 = Operation(lambda x: np.exp(x), lambda x  : np.exp(x))

op3 = Operation(lambda x,w: x*w,     lambda x,w: w,
                 0.01, [2], [lambda x,w: x])
```

Python Demo

- Example: $y=Ae^{Bx}$

op1: $y=Bx$	$y'=B$	$Y'= x$
op2: $y=e^x$	$y'= e^x$	
op3: $y=Ax$	$y'=A$	$Y'= x$

```
op1 = Operation(lambda x,w: x*w,     lambda x,w: w,
                0.01, [0.2], [lambda x,w: x])

op2 = Operation(lambda x: np.exp(x), lambda x  : np.exp(x))

op3 = Operation(lambda x,w: x*w,     lambda x,w: w,
                0.01, [2], [lambda x,w: x])
```

Python Demo

- Example: $y=Ae^{Bx}$

op1: $y=Bx$	$y'=B$	$Y'=x$
op2: $y=e^x$	$y'=e^x$	
op3: $y=Ax$	$y'=A$	$Y'=x$

```
op1 = Operation(lambda x,w: x*w,     lambda x,w: w,
                 0.01, [0.2], [lambda x,w: x])

op2 = Operation(lambda x: np.exp(x), lambda x  : np.exp(x))

op3 = Operation(lambda x,w: x*w,     lambda x,w: w,
                 0.01, [2], [lambda x,w: x])
```

Python Demo

- Example: $y=Ae^{Bx}$

op1: $y=Bx$	$y'=B$	$Y'= x$
op2: $y=e^x$	$y'= e^x$	
op3: $y=Ax$	$y'=A$	$Y'= x$

```
op1 = Operation(lambda x,w: x*w,     lambda x,w: w,
                0.01, [0.2], [lambda x,w: x])

op2 = Operation(lambda x: np.exp(x), lambda x  : np.exp(x))

op3 = Operation(lambda x,w: x*w,     lambda x,w: w,
                0.01, [2], [lambda x,w: x])
```

Python Demo

- Example: $y=Ae^{Bx}$

op1: $y=Bx$	$y'=B$	$Y'= x$
op2: $y=e^x$	$y'= e^x$	
op3: $y=Ax$	$y'=A$	$Y'= x$

```
p = Path([op1,op2,op3],in,label)
```

Radar challenge

Weather radar

- High resolution
- 360 deg field around radar
- Difficult to calibrate
- Readings quantized into 34 bins (including 0-bin)

Rain gauge

- Spot measurements
- More accurate

Radar challenge

Task: Use rain gauge measurements to improve radar calibration

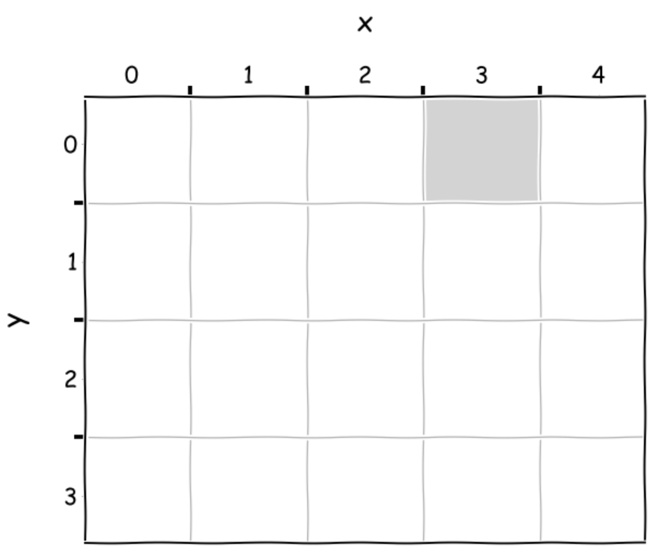
Radar challenge

- Rain gauge dataset
  - CSV format
  - Rows: weekly values
  - Columns: 50 weather measurement stations
    (numbered 0 to 49, stations are in SG)
  - Check for null values!

	A	B	C	D	E	F
1	Year	Week	0	1	2	3
2	2017	1	17.2	15.6	12.6	4.2
3	2017	2	0.2	2.4	2	26.8
4	2017	3	68.8	21.2	54.4	67.2
5	2017	4	64.4	106	81.2	70.6
6	2017	5	2.4	7.4	13.8	11.2
7	2017	6	6.4	79.8	47.4	30.4
8	2017	7	3	12.6	6.2	19.6
9	2017	8	26.4	27.2	14.4	29.6
10	2017	9	40.2	45.2	25.6	31.2
11	2017	10	10.2	33.2	8.8	36.2

Radar challenge

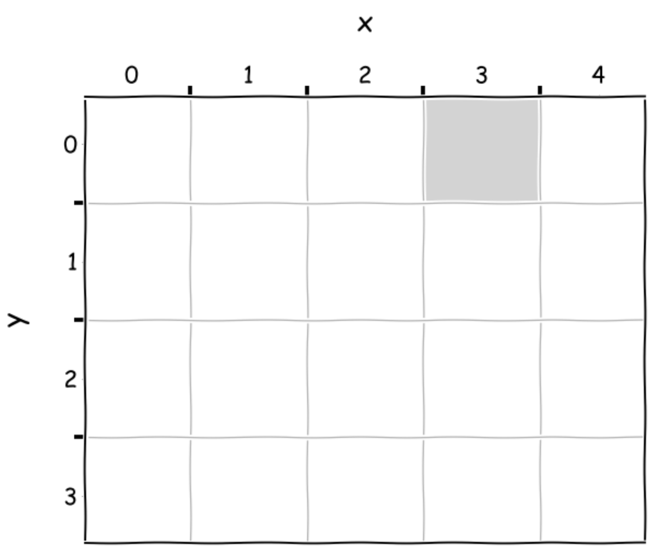
- Radar dataset
  - CSV format
  - Each week has its own file: no file = data gap!
  - Rows: radar coordinate (480 x 480 grid)
  - Columns: counts by bin number (34 bins)



y	x	0	1	2	3
0	0	1917	0	0	2
0	1	1919	0	0	0
0	2	1918	0	2	2
0	3	1919	0	0	0
0	4	1919	0	0	0
0	5	1918	1	0	0
0	6	1919	0	2	0
0	7	1924	0	0	0
0	8	1918	0	0	2
0	9	1923	0	0	1
0	10	1923	0	0	0
0	11	1928	0	0	0
0	12	1924	0	0	3

Radar challenge

- Radar dataset
    - (0,0) coordinate is at 1.980 deg latitude, 103.338 deg longitude
    - Distance between grid points: 292m
    - Radar snapshot interval: 5 mins



y	x	0	1	2	3
0	0	1917	0	0	2
0	1	1919	0	0	0
0	2	1918	0	2	2
0	3	1919	0	0	0
0	4	1919	0	0	0
0	5	1918	1	0	0
0	6	1919	0	2	0
0	7	1924	0	0	0
0	8	1918	0	0	2
0	9	1923	0	0	1
0	10	1923	0	0	0
0	11	1928	0	0	0
0	12	1924	0	0	3

Radar challenge

- Suggested starting model

$$r(x, y) = A\, e^{B \cdot k(x,y)}$$

x, y	: location coordinates
$r(x, y)$	: rainfall rate at location (x, y)
$k(x, y)$	: bin number of radar signal at (x, y) range of values – 0,1,…,33
A, B	: model parameters

$$\boxed{A = 0.079\,, \qquad B = 0.228}$$

Radar challenge

- Required outcomes
  1. Determine the locations of the 50 weather stations in radar coordinates
  2. Improved radar model
     - Use suitable performance metrics
  3. Written report & slide deck
     - Explain and substantiate methodology
     - Describe model used and show improvement over ‘simple’ model
     - Attach code in appendix

<u>There are no restrictions on the methods or tools used for this competition</u>

Good Luck!