Introduction to Nonlinear Statistics and Neural Networks

Vladimir Krasnopolsky

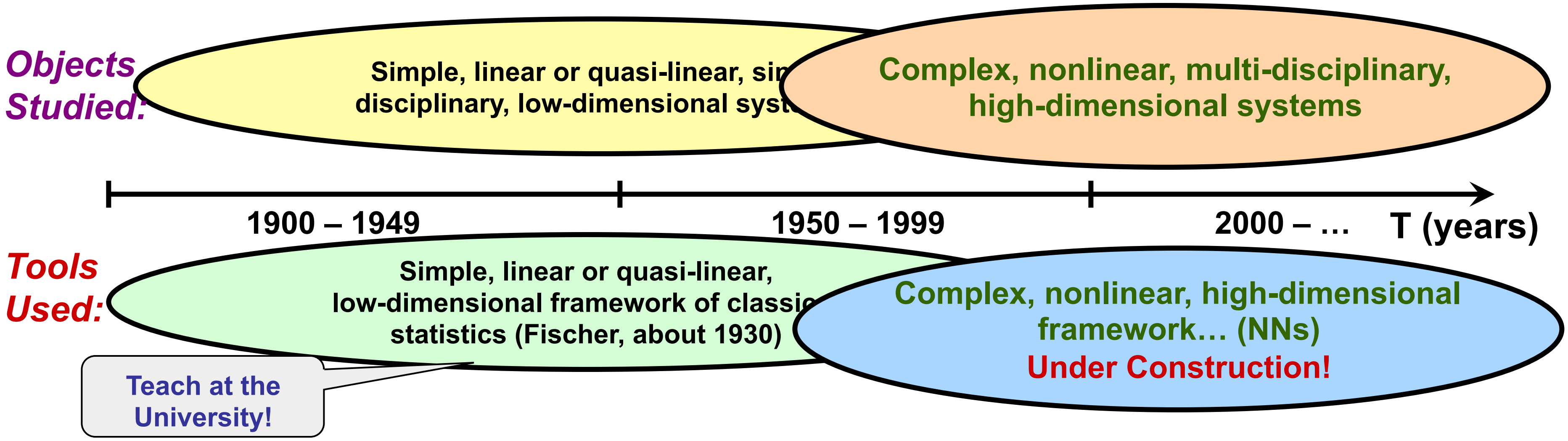
NCEP/NOAA & ESSIC/UMD

http://polar.ncep.noaa.gov/mmab/people/kvladimir.html

Outline

- Introduction: Regression Analysis
- Regression Models (Linear & Nonlinear)
- NN Tutorial
- Some Atmospheric & Oceanic Applications
  - Accurate and fast emulations of model physics
  - NN Multi-Model Ensemble
- How to Apply NNs
- Conclusions

Evolution in Statistics



Objects Studied:

Simple, linear or quasi-linear, sin
disciplinary, low-dimensional syst

Complex, nonlinear, multi-disciplinary, high-dimensional systems

1900 – 1949

1950 – 1999

2000 – …

T (years)

Tools Used:

Simple, linear or quasi-linear, low-dimensional framework of classic statistics (Fischer, about 1930)

Complex, nonlinear, high-dimensional framework… (NNs)

Under Construction!

Teach at the University!

- **Problems for Classical Paradigm:**
  - **Nonlinearity & Complexity**
  - **High Dimensionality -** ***Curse of Dimensionality***
- **New Paradigm under Construction:**
  - **Is still quite fragmentary**
  - **Has many different names and gurus**
  - **NNs are one of the tools developed inside this paradigm**

Statistical Inference:
A Generic Problem

Problem:

Information exists in the form of finite sets of values of several ***related variables*** (**sample** or **training set**) – a part of the **population**:

$$\aleph = \{(x_1, x_2, \ldots, x_n)_p, z_p\}_{p=1,2,\ldots,N}$$

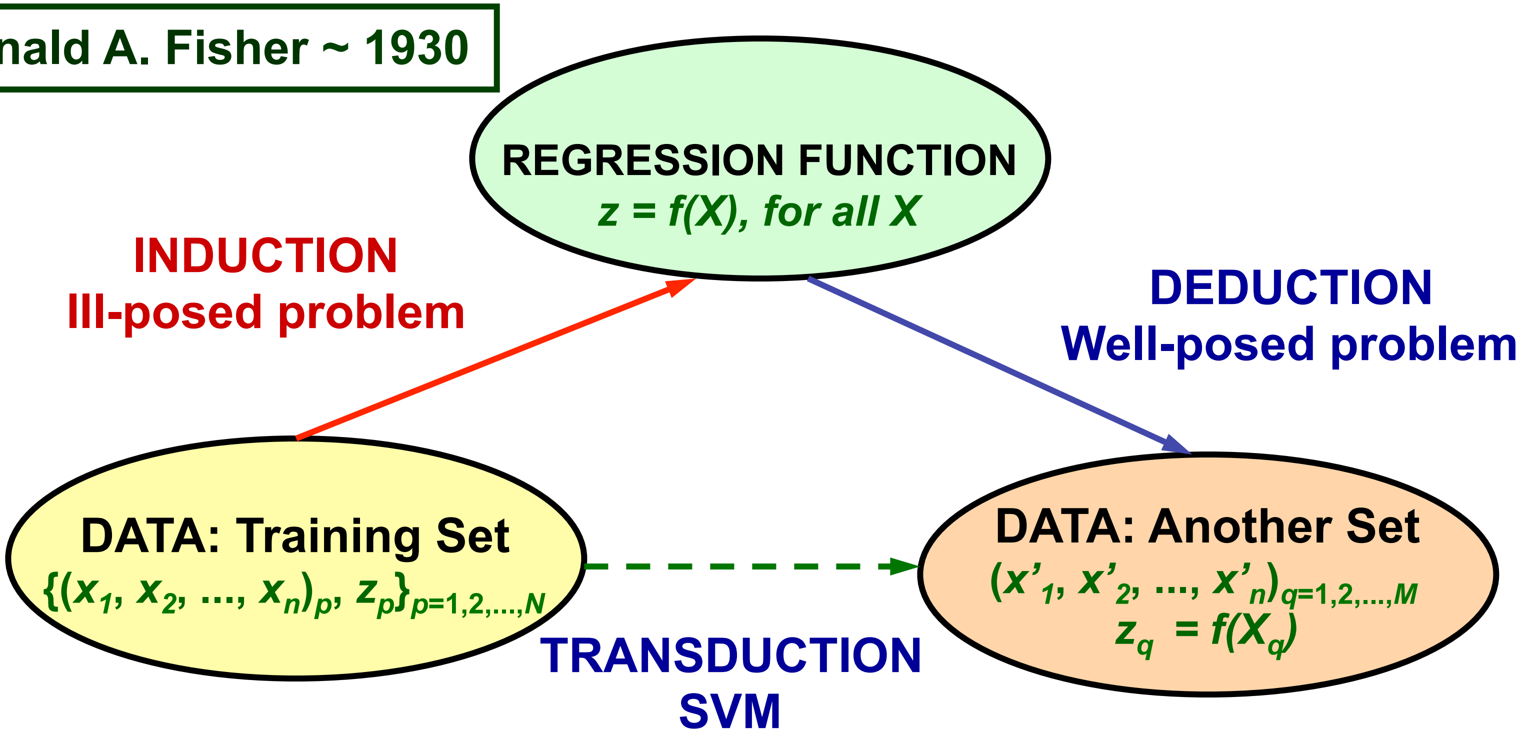
- $x_1, x_2, \ldots, x_n$ - independent variables (accurate),
- z - response variable (may contain observation errors ε)

We want to find responses z'_q for another set of independent variables $\aleph' = \{(x'_1, x'_2, \ldots, x'_n)_q\}_{q=1,..,M}$

$$\aleph' \notin \aleph$$

Regression Analysis (1):
General Solution and Its Limitations

Sir Ronald A. Fisher ~ 1930



REGRESSION FUNCTION

$z = f(X)$, for all X

INDUCTION

Ill-posed problem

DEDUCTION

Well-posed problem

DATA: Training Set

$\{(x_1, x_2, ..., x_n)_p, z_p\}_{p=1,2,...,N}$

TRANSDUCTION

SVM

DATA: Another Set

$(x'_1, x'_2, ..., x'_n)_{q=1,2,...,M}$

$z_q = f(X_q)$

Find mathematical function *f* which describes this relationship:

1. **Identify the unknown function *f***
2. **Imitate or emulate the unknown function *f***

Regression Analysis (2):
A Generic Solution

- The effect of *independent variables* on the *response* is expressed mathematically by the *regression or response function f:*

$$y = f(x_1, x_2, \ldots, x_n; a_1, a_2, \ldots, a_q)$$

- y - dependent variable
- $a_1, a_2, \ldots, a_q$ - regression parameters (unknown!)
- f - the form is usually assumed to be known
- Regression model for observed response variable:

$$z = y + \varepsilon = f(x_1, x_2, \ldots, x_n; a_1, a_2, \ldots, a_q) + \varepsilon$$

- ε - error in observed value z

Regression Models (1):
Maximum Likelihood

- Fischer suggested to determine unknown regression parameters $\{a_i\}_{i=1,..,q}$ maximizing the functional:

$$L(a) = \sum_{p=1}^{N} \ln[\rho(z_p - y_p)]; \quad where\ y_p = f(x_p$$

Not always!!!

here $\rho(\varepsilon)$ is the probability density function of errors ε_i

- *In a case* when $\rho(\varepsilon)$ is a *normal distribution*

$$\rho(z-y) = \alpha \cdot \mathbf{exp}(-\frac{(z-y)^2}{\sigma^2})$$

the maximum likelihood => least squares

$$L(a) = \sum_{p=1}^{N} \mathbf{ln}\left[\alpha \cdot \mathbf{exp}(-\frac{(z_p - y_p)^2}{\sigma^2})\right] = A - B \cdot \sum_{p=1}^{N} (z_p - y_p)^2$$

$$\mathbf{max}\, L \Rightarrow \mathbf{min} \sum_{p=1}^{N} (z_p - y_p)^2$$

Regression Models (2):
Method of Least Squares

- To find ***unknown regression parameters*** $\{a_i\}_{i=1,2,\ldots,q}$, the ***method of least squares*** can be applied:

$$E(a_1, a_2, \ldots, a_q) = \sum_{p=1}^{N} (z_p - y_p)^2 = \sum_{p=1}^{N} [z_p - f((x_1, \ldots, x_n)_p; a_1, a_2, \ldots, a_q)]^2$$

- ***E(a***$E(a_1,\ldots,a_q)$ - **error function** = the sum of squared deviations.
- To estimate $\{a_i\}_{i=1,2,\ldots,q}$ => **minimize** ***E*** => solve the system of equations:

$$\frac{\partial E}{\partial a_i} = 0; \quad i = 1,2,\ldots,q$$

- **Linear** and **nonlinear** cases.

Regression Models (3):
Examples of Linear Regressions

- **Simple Linear Regression:**

  $z = a_0 + a_1 x_1 + \varepsilon$

- **Multiple Linear Regression:**

  $z = a_0 + a_1 x_1 + a_2 x_2 + \ldots + \varepsilon = a_0 + \sum_{i=1}^{n} a_i x_i + \varepsilon$

- **Generalized Linear Regression:**

  $z = a_0 + a_1 f_1(x_1) + a_2 f_2(x_2) + \ldots + \varepsilon = a_0 + \sum_{i=1}^{n} a_i f_i(x_i) + \varepsilon$

  No free parameters

  – **Polynomial regression, $f_i(x) = x^i$,**

  $z = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots + \varepsilon$

  – **Trigonometric regression, $f_i(x) = \cos(ix)$**

  $z = a_0 + a_1 \cos(x) + a_1 \cos(2\,x) + \ldots + \varepsilon$

Regression Models (4):

Examples of Nonlinear Regressions

- **Response Transformation Regression:**

$$G(z) = a_0 + a_1 x_1 + \varepsilon$$

- **Example:**

$$z = \exp(a_0 + a_1 x_1)$$

$$G(z) = \ln(z) = a_0 + a_1 x_1$$

- **Projection-Pursuit Regression:**

$$y = a_0 + \sum_{j=1}^{k} a_j f(\sum_{i=1}^{n} \Omega_{ji} x_i)$$

Free nonlinear parameters

- **Example:**

$$z = a_0 + \sum_{j=1}^{k} a_j \tanh(b_j + \sum_{i=1}^{n} \Omega_{ji} x_i) + \varepsilon$$

NN Tutorial:
Introduction to Artificial NNs

- NNs as Continuous Input/Output Mappings
  - Continuous Mappings: definition and some examples
  - NN Building Blocks: neurons, activation functions, layers
  - Some Important Theorems
- NN Training
- Major Advantages of NNs
- Some Problems of Nonlinear Approaches

Mapping

Generalization of Function

- **Mapping:** **A** *rule of correspondence established between vectors* **in vector spaces** $\Re^n$ **and** $\Re^m$ **that associates each vector** *X* **of a vector space** $\Re^n$ **with a vector** *Y* **in another vector space** $\Re^m$ **.**

$$\left.\begin{array}{l} Y = F(X) \\ X = \{x_1, x_2, \ldots, x_n\}, \in \Re^n \\ Y = \{y_1, y_2, \ldots, y_m\}, \in \Re^m \end{array}\right\} \neq \begin{bmatrix} y_1 = f_1(x_1, x_2, \ldots, x_n) \\ y_2 = f_2(x_1, x_2, \ldots, x_n) \\ \vdots \\ y_m = f_m(x_1, x_2, \ldots, x_n) \end{bmatrix}$$

Mapping $Y = F(X)$: examples

- **Time series prediction:**

  $X = \{x_t, x_{t-1}, x_{t-2}, \ldots, x_{t-n}\}$, - Lag vector

  $Y = \{x_{t+1}, x_{t+2}, \ldots, x_{t+m}\}$ - Prediction vector

  (Weigend & Gershenfeld, "Time series prediction", 1994)

- **Calculation of precipitation climatology:**

  X = {*Cloud parameters, Atmospheric parameters*}

  Y = {*Precipitation climatology*}

  (Kondragunta & Gruber, 1998)

- **Retrieving surface wind speed over the ocean from satellite data (SSM/I):**

  X = {*SSM/I brightness temperatures*}

  Y = {*W, V, L, SST*}

  (Krasnopolsky, et al., 1999; operational since 1998)

- **Calculation of long wave atmospheric radiation:**

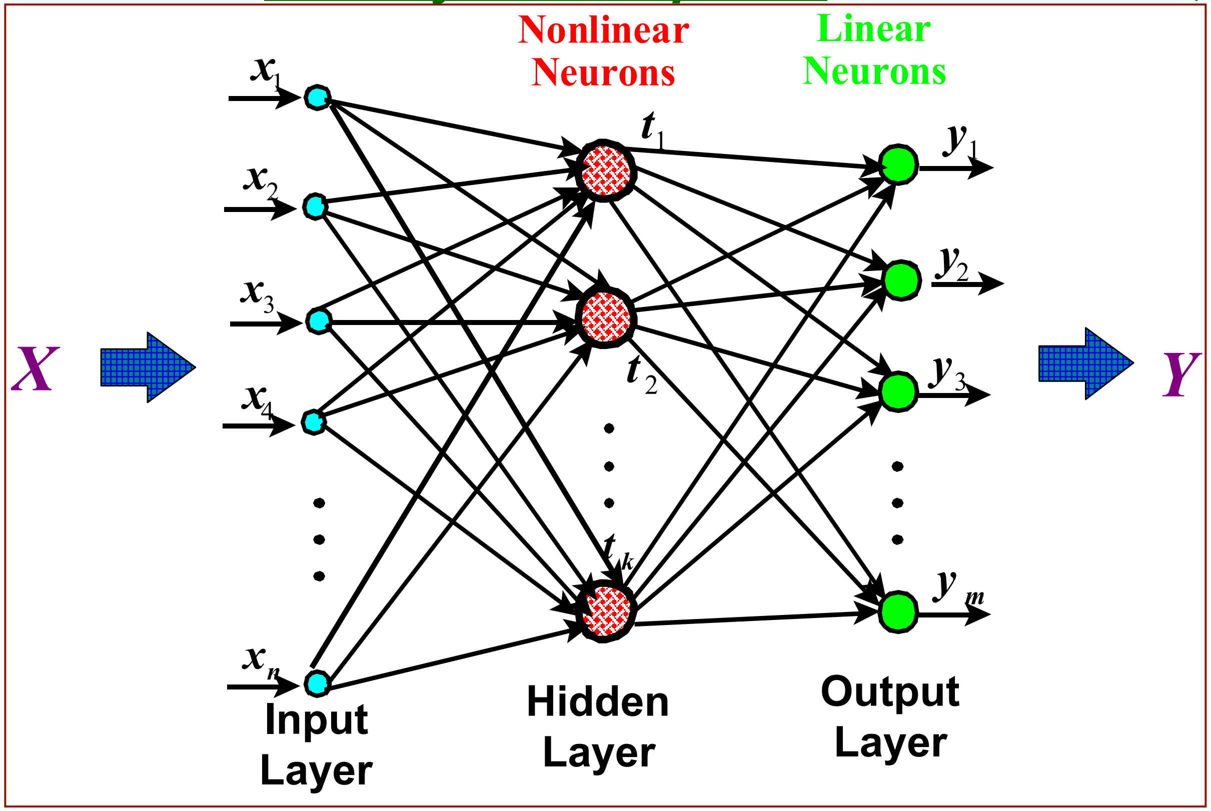
  X = {*Temperature, moisture, O_3, CO_2, cloud parameters profiles, surface fluxes, etc.*}

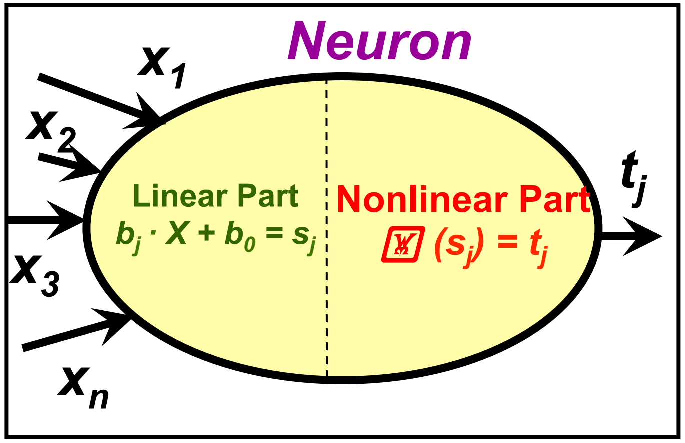
  Y = {*Heating rates profile, radiation fluxes*}

  (Krasnopolsky et al., 2005)

NN - Continuous Input to Output Mapping

Multilayer Perceptron: **Feed Forward, Fully Connected**





$$t_j = \phi(b_{j0} + \sum_{i=1}^{n} b_{ji} \cdot x_i) =$$

$$= \tanh(b_{j0} + \sum_{i=1}^{n} b_{ji} \cdot x_i)$$

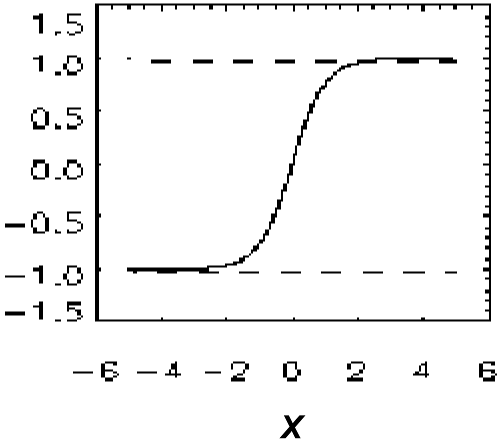
Y = F$_{NN}$(X)

Jacobian !

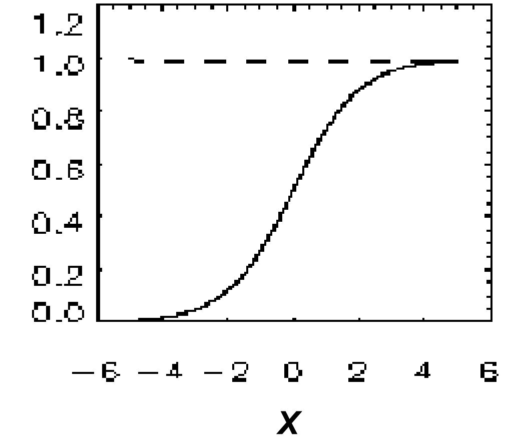
$$\begin{cases} y_q = a_{q0} + \sum_{j=1}^{k} a_{qj} \cdot t_j = a_{q0} + \sum_{j=1}^{k} a_{qj} \cdot \phi(b_{j0} + \sum_{i=1}^{n} b_{ji} \cdot x_i) = \\ = a_{q0} + \sum_{j=1}^{k} a_{qj} \cdot \tanh(b_{j0} + \sum_{i=1}^{n} b_{ji} \cdot x_i); \quad q = 1,2,\ldots,m \end{cases}$$

Some Popular Activation Functions

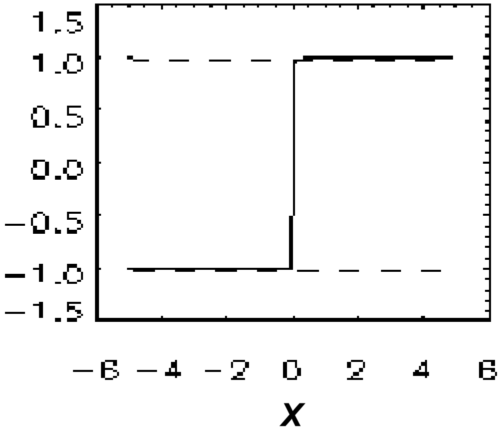
tanh(x)



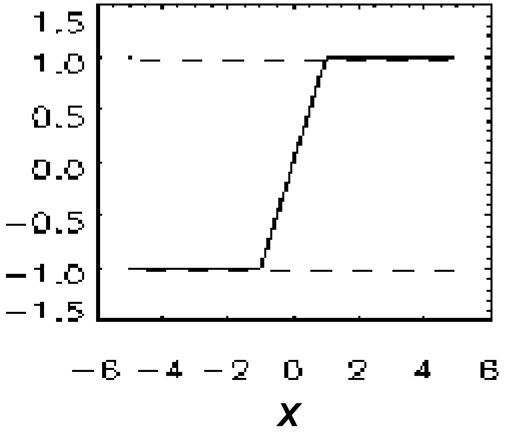
Sigmoid, $(1 + exp(-x))^{-1}$



Hard Limiter



Ramp Function



NN as a Universal Tool for Approximation of Continuous & Almost Continuous Mappings

Some Basic Theorems:

- **Any function or mapping *$Z = F(X)$*, continuous on a compact subset, *can be approximately represented by* a p (p ⊠ 3) layer *NN in the sense of uniform convergence* (e.g., Chen & Chen, 1995; Blum and Li, 1991, Hornik, 1991; Funahashi, 1989, etc.)**

- **The** error bounds **for the uniform approximation on compact sets (Attali & Pagès, 1997):**

$$||Z - Y|| = ||F(X) - F_{NN}(X)|| \sim C/k$$

k **-number of neurons in the hidden layer**

***C* – does not depend on *n* (avoiding *Curse of Dimensionality!*)**

NN training (1)

- For the mapping *Z = F (X)* create a ***training set*** - set of matchups $\{X_i, Z_i\}_{i=1,\ldots,N}$, where X_i is ***input vector*** and Z_i - ***desired output vector***
- Introduce ***an error or cost function E***:

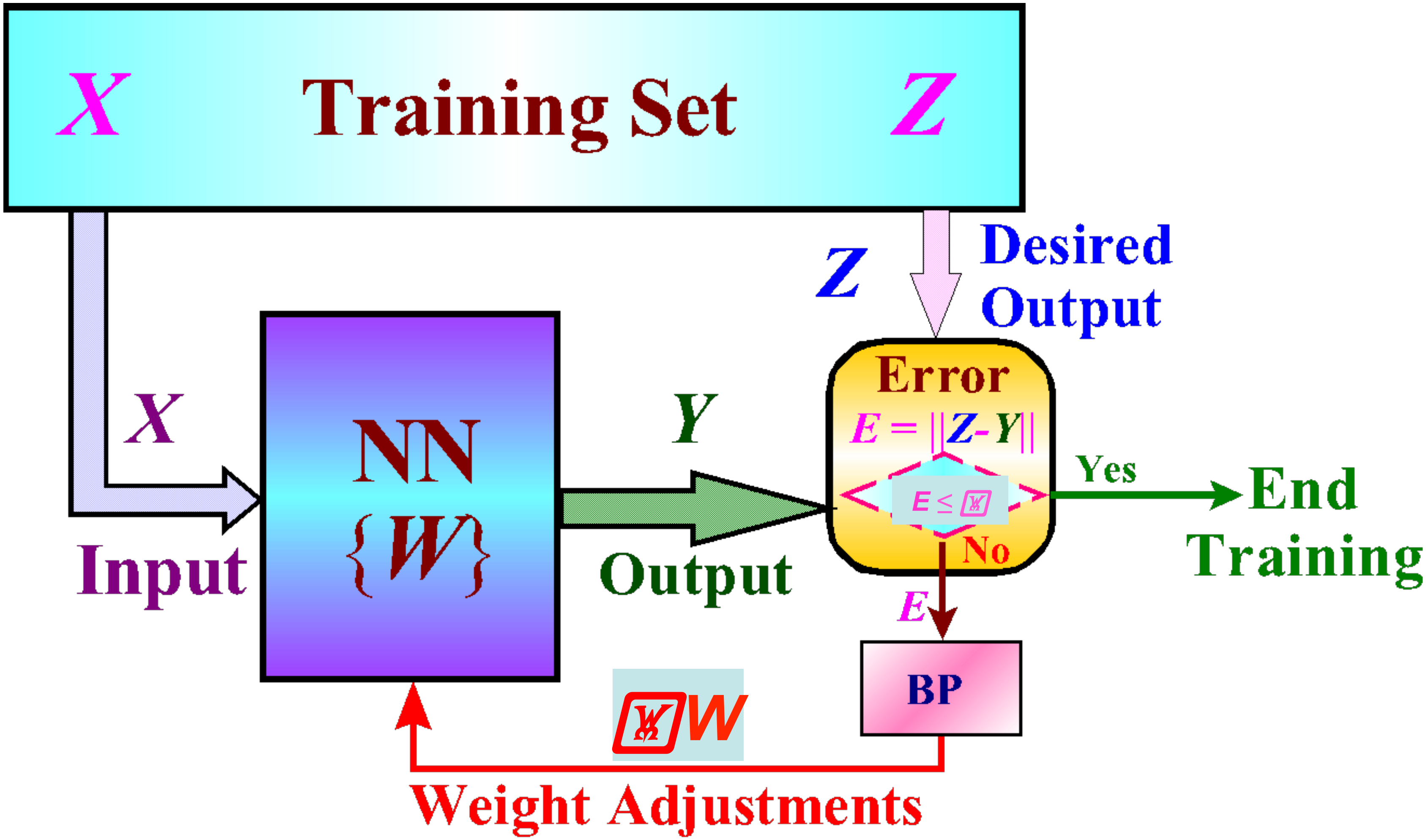
$$E(a,b) = ||Z - Y|| = \sum_{i=1}^{N} \left| Z_i - F_{NN}(X_i) \right|^2 ,$$

where $Y = F_{NN}(X)$ is neural network

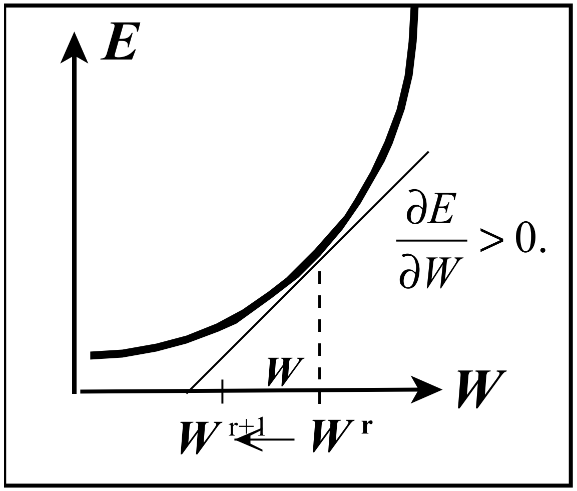
- Minimize the cost function: *min*{*E*(*a*,*b*)} and find optimal weights (a_0, b_0)
- Notation: *W* = {*a*, *b*} - all weights.

NN Training (2)

One Training Iteration



Backpropagation (BP) Training Algorithm


E
$\frac{\partial E}{\partial W} > 0.$
W
W
$W^{r+1} \leftarrow W^{r}$

- **BP is a simplified steepest descent:**

$$\Delta W = -\eta \frac{\partial E}{\partial W}$$

where *W* - any weight, *E* - error function,

η - learning rate, and ΔW - weight increment

- **Derivative can be calculated analytically:**

$$\frac{\partial E}{\partial W} = -2\sum_{i=1}^{N}[Z_i - F_{NN}(X_i)]\cdot\frac{\partial F_{NN}(X_i)}{\partial W}$$

- **Weight adjustment after r-th iteration:**

$$W^{r+1} = W^{r} + \Delta W$$

- **BP training algorithm is robust but slow**

Generic Neural Network

FORTRAN Code:

```
DATA W1/.../, W2/.../, B1/.../, B2/.../, A/.../, B/.../ !  Task specific part
!=====================================================
DO K = 1,OUT
!
  DO I = 1, HID
     X1(I) = tanh(sum(X * W1(:,I) + B1(I))
  ENDDO !  I
!
  X2(K) = tanh(sum(W2(:,K)*X1) + B2(K))
  Y(K) = A(K) * X2(K) + B(K)
! --
  XY = A(K) * (1. -X2(K) * X2(K))
  DO J = 1, IN
     DUM = sum((1. -X1 * X1) * W1(J,:) * W2(:,K))
     DYDX(K,J) = DUM * XY
  ENDDO !  J
!
ENDDO !  K
```

NN Output

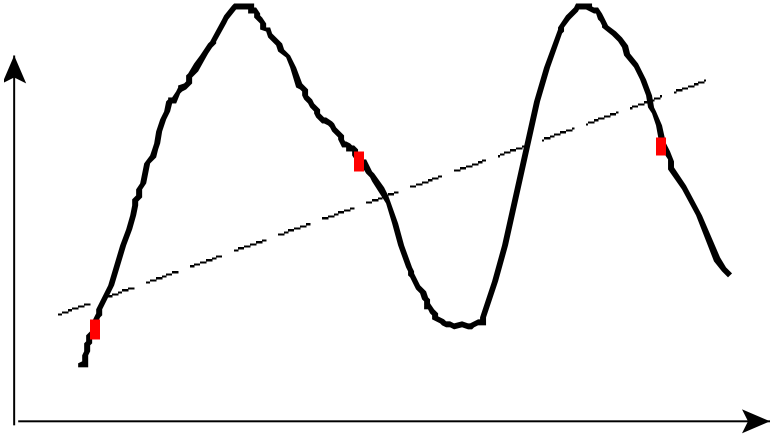
Jacobian

Major Advantages of NNs :

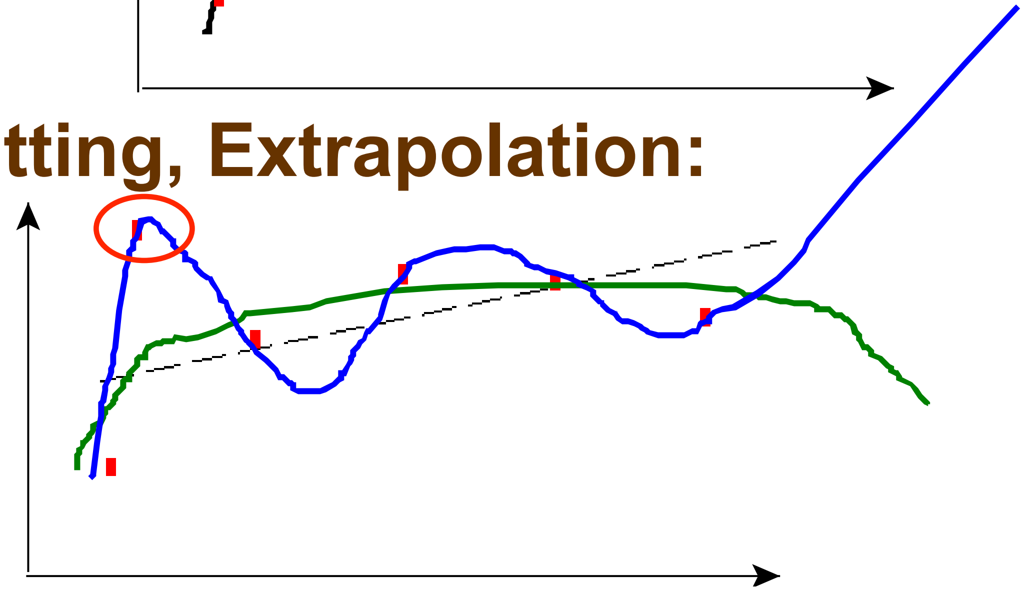
- NNs are very ***generic, accurate and convenient*** mathematical (statistical) models which are able **to emulate numerical model components**, which are complicated nonlinear input/output relationships (continuous or almost continuous mappings).
- NNs avoid ***Curse of Dimensionality***
- NNs are *robust* with respect to random noise and fault-tolerant.
- NNs are ***analytically differentiable*** (training, error and sensitivity analyses): **almost free Jacobian!**
- NNs emulations are accurate and fast but NO FREE LUNCH!
- Training is complicated and time consuming nonlinear optimization task; ***however, training should be done only once for a particular application!***
- Possibility of online adjustment
- NNs are **well-suited for parallel and vector processing**

NNs & Nonlinear Regressions: Limitations (1)

- Flexibility and Interpolation:



- Overfitting, Extrapolation:



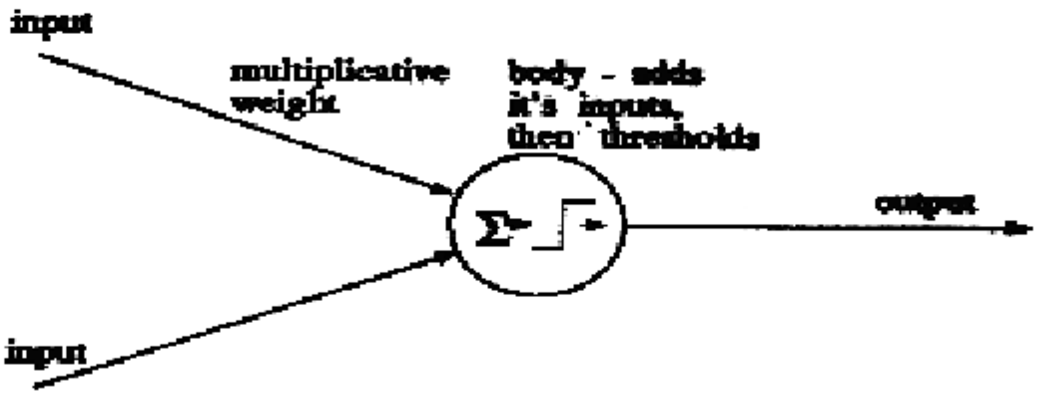
NNs & Nonlinear Regressions: Limitations (2)

- Consistency of estimators: α is a consistent estimator of parameter A, if $\alpha \to A$ as the size of the sample $n \to N$, where N is the size of the population.
- For NNs and Nonlinear Regressions consistency can be usually “proven” only numerically.
- Additional independent data sets are required for test (demonstrating consistency of estimates).

ARTIFICIAL NEURAL NETWORKS:
BRIEF HISTORY

- **1943 - McCulloch and Pitts introduced a model of the neuron**

Modeling the single neuron

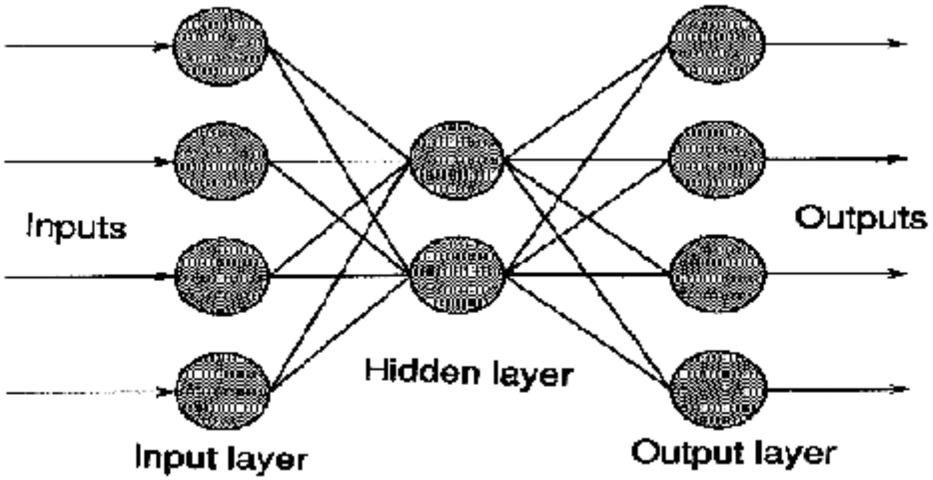


- **1962 - Rosenblat introduced the one layer "perceptrons", the model neurons, connected up in a simple fashion.**
- **1969 - Minsky and Papert published the book which practically “closed the field”**

ARTIFICIAL NEURAL NETWORKS:
BRIEF HISTORY

- 1986 - Rumelhart and McClelland proposed the "multilayer perceptron" (MLP) and showed that it is a perfect application for parallel distributed processing.

The multilayer perceptron



- From the end of the 80's there has been explosive growth in applying NNs to various problems in different fields of science and technology

Atmospheric and Oceanic NN Applications

- Satellite Meteorology and Oceanography
  - Classification Algorithms
  - Pattern Recognition, Feature Extraction Algorithms
  - Change Detection & Feature Tracking Algorithms
  - Fast Forward Models for Direct Assimilation
  - Accurate Transfer Functions (Retrieval Algorithms)
- Predictions
  - Geophysical time series
  - Regional climate
  - Time dependent processes
- NN Ensembles
  - Fast NN ensemble
  - Multi-model NN ensemble
  - NN Stochastic Physics
- Fast NN Model Physics
- Data Fusion & Data Mining
- Interpolation, Extrapolation & Downscaling
- Nonlinear Multivariate Statistical Analysis
- Hydrological Applications

Developing Fast NN Emulations for Parameterizations of Model Physics

Atmospheric Long & Short Wave Radiations

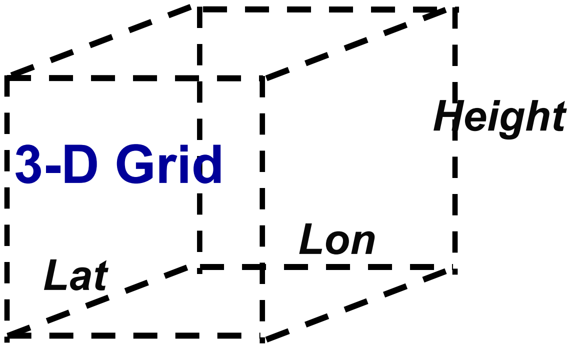
General Circulation Model

The set of conservation laws (mass, energy, momentum, water vapor, ozone, etc.)

- ***First Priciples/Prediction 3-D Equations on the Sphere:***

$$\frac{\partial \psi}{\partial t} + D(\psi, x) = P(\psi, x)$$

  - **ψ - a 3-D prognostic/dependent variable, e.g., temperature**
  - ***x*** **- a 3-D independent variable: *x, y, z* & *t***
  - ***D*** **- dynamics (spectral or gridpoint)**
  - ***P*** **- physics or parameterization of physical processes (1-D vertical r.h.s. forcing)**
- ***Continuity Equation***
- ***Thermodynamic Equation***
- ***Momentum Equations***



General Circulation Model

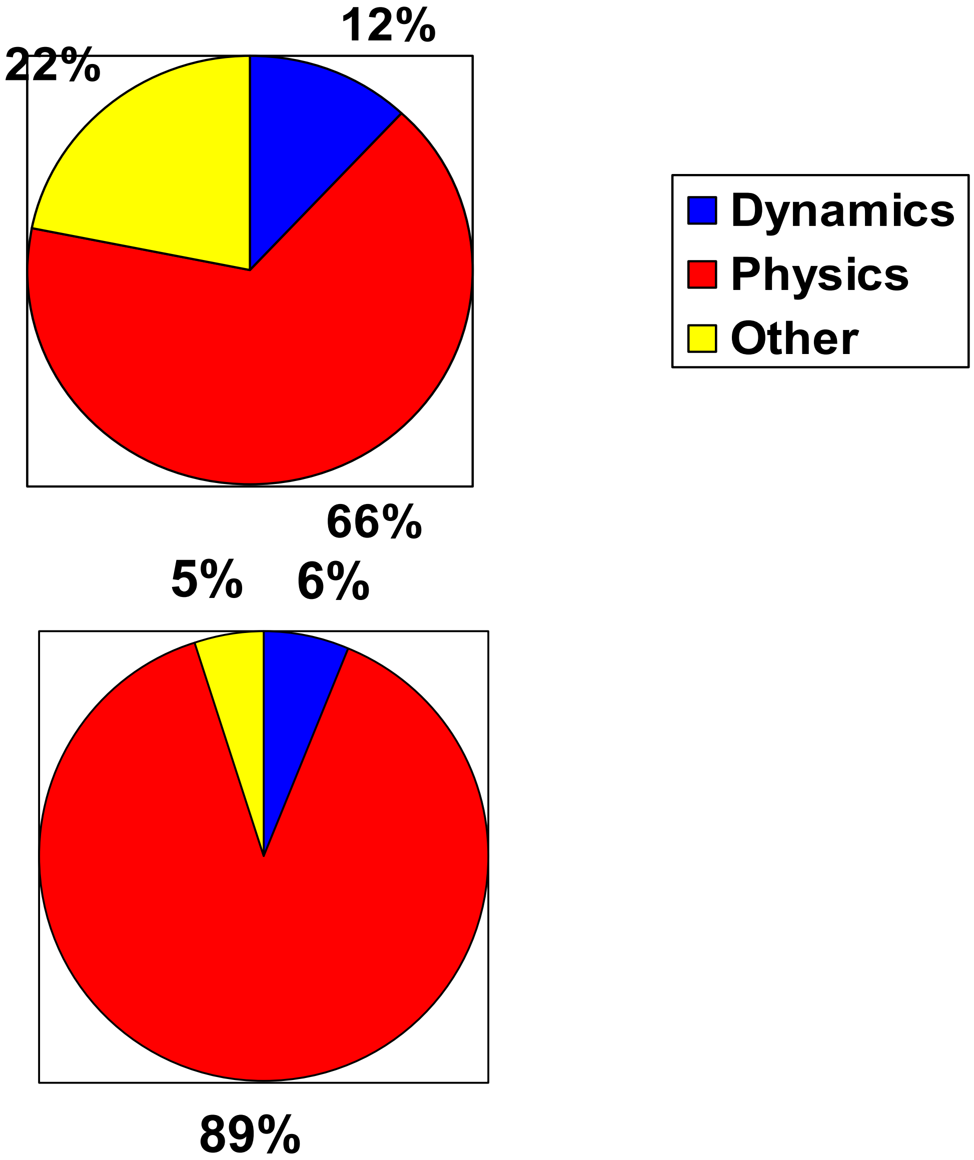
Physics – P, represented by 1-D (vertical) parameterizations

- Major components of *P* = {*R, W, C, T, S*}:
  - *R* - radiation (long & short wave processes)
  - *W* – convection, and large scale precipitation processes
  - C - clouds
  - *T* – turbulence
  - *S* – surface model (land, ocean, ice – air interaction)
- Each component of *P* is a 1-D parameterization of complicated set of multi-scale theoretical and empirical physical process models ***simplified for computational reasons***
- P is the ***most time consuming*** part of GCMs*!*

Distribution of Total Climate Model Calculation Time

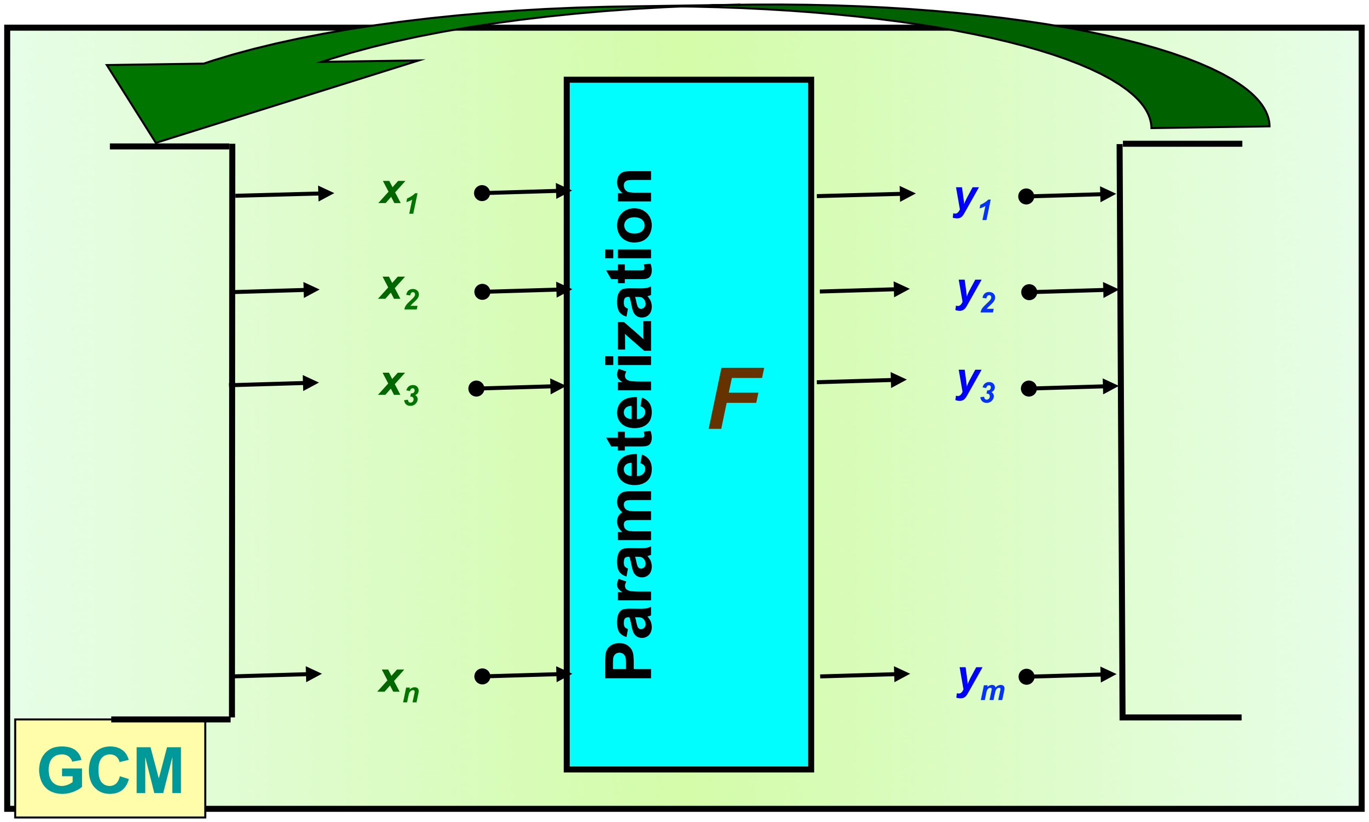
Current NCAR Climate Model (T42 x L26): □ 3□ x 3.5□

Near-Term Upcoming Climate Models (estimated) : □ 1□ x 1□



Generic Situation in Numerical Models

Parameterizations of Physics are Mappings

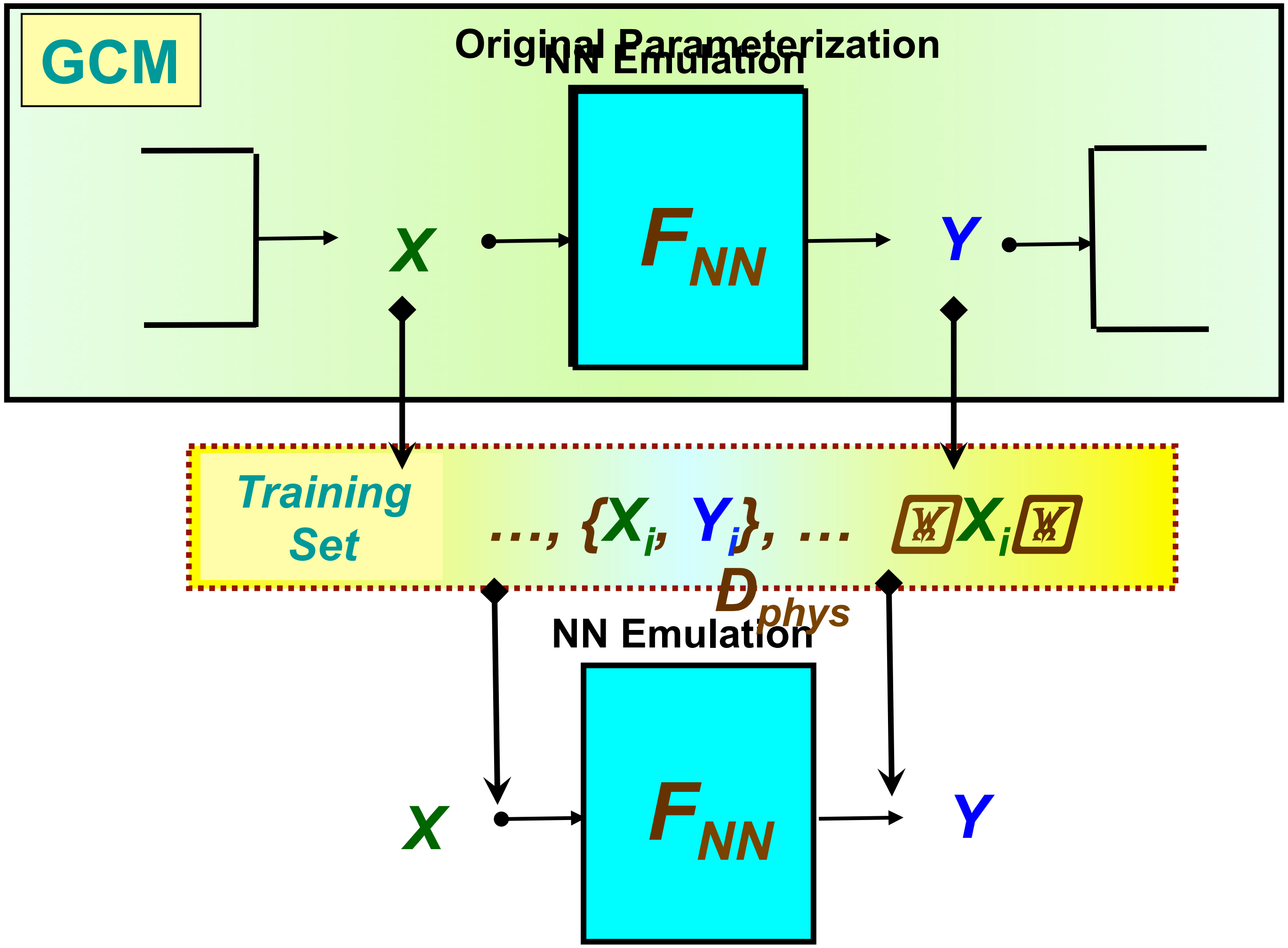


$$Y=F(X)$$

Generic Solution – “NeuroPhysics”

Accurate and Fast NN Emulation for Physics Parameterizations

Learning from Data



NN for NCAR CAM Physics

CAM Long Wave Radiation

- **Long Wave Radiative Transfer*:***

$$F^{\downarrow}(p) = B(p_t)\cdot\varepsilon(p_t, p) + \int_{p_t}^{p}\alpha(p_t, p)\cdot dB(p')$$

$$F^{\uparrow}(p) = B(p_s) - \int_{p}^{p_s}\alpha(p, p')\cdot dB(p')$$

$$B(p) = \sigma\cdot T^4(p) \quad -the\ Stefan - Boltzman\ relation$$

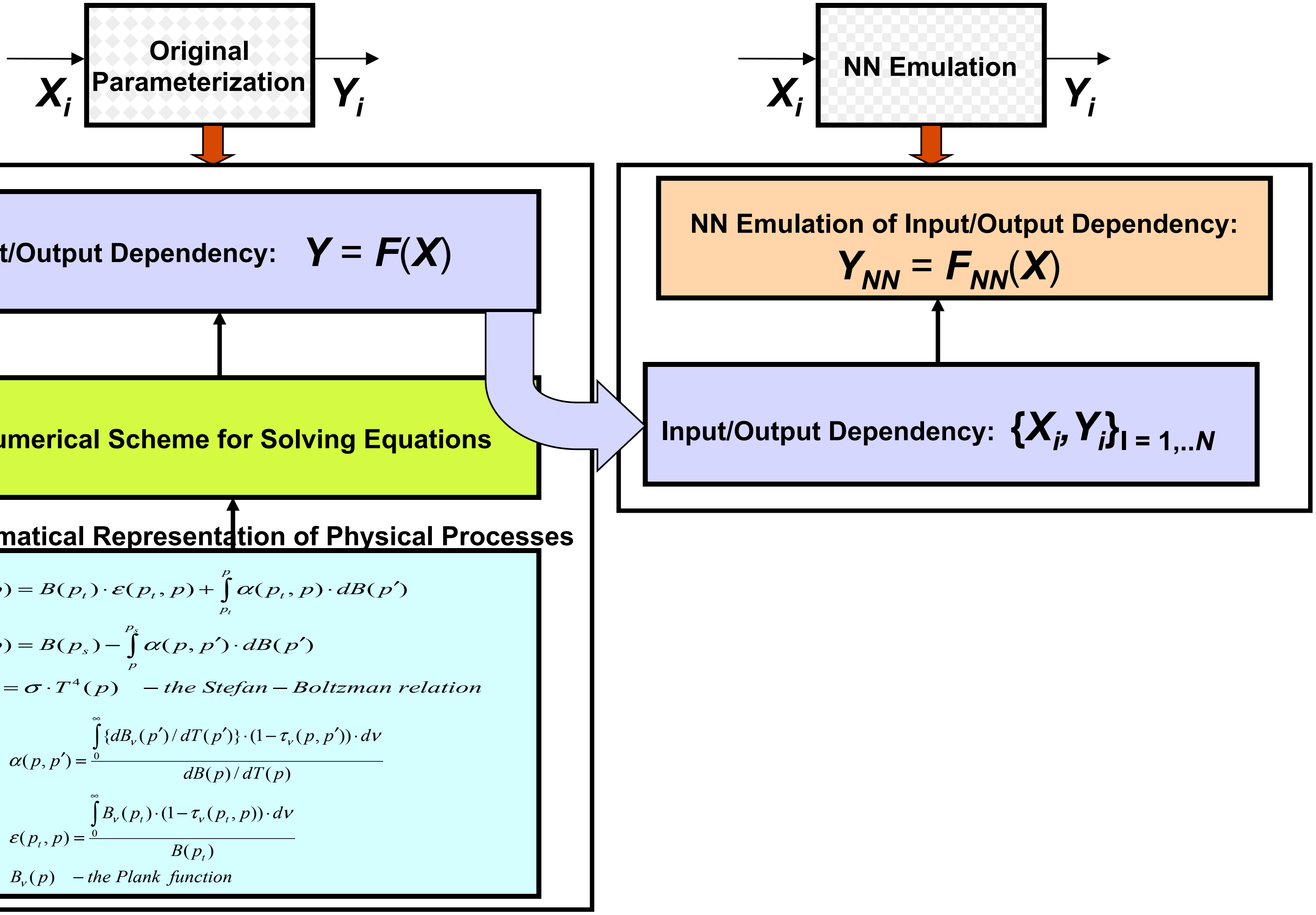
- **Absorptivity & Emissivity (optical properties):**

$$\alpha(p, p') = \frac{\int_0^{\infty}\{dB_{\nu}(p')/dT(p')\}\cdot(1-\tau_{\nu}(p, p'))\cdot d\nu}{dB(p)/dT(p)}$$

$$\varepsilon(p_t, p) = \frac{\int_0^{\infty}B_{\nu}(p_t)\cdot(1-\tau_{\nu}(p_t, p))\cdot d\nu}{B(p_t)}$$

$$B_{\nu}(p) \quad -the\ Plank\ \ function$$

The Magic of NN Performance


X_i
Original Parameterization
Y_i
X_i
NN Emulation
Y_i
Input/Output Dependency: $Y = F(X)$
Numerical Scheme for Solving Equations
Mathematical Representation of Physical Processes
NN Emulation of Input/Output Dependency: $Y_{NN} = F_{NN}(X)$
Input/Output Dependency: $\{X_i, Y_i\}_{I = 1,..N}$

$$F^{\downarrow}(p) = B(p_t)\cdot\varepsilon(p_t, p) + \int_{p_t}^{p} \alpha(p_t, p)\cdot dB(p')$$

$$F^{\uparrow}(p) = B(p_s) - \int_{p}^{p_s} \alpha(p, p')\cdot dB(p')$$

$$B(p) = \sigma\cdot T^4(p) \quad - the\ Stefan - Boltzman\ relation$$

$$\alpha(p, p') = \frac{\int_0^{\infty}\{dB_\nu(p')/dT(p')\}\cdot(1-\tau_\nu(p, p'))\cdot d\nu}{dB(p)/dT(p)}$$

$$\varepsilon(p_t, p) = \frac{\int_0^{\infty} B_\nu(p_t)\cdot(1-\tau_\nu(p_t, p))\cdot d\nu}{B(p_t)}$$

$$B_\nu(p) \quad - the\ Plank\ function$$

Neural Networks for NCAR (NCEP) LW Radiation

NN characteristics

- **220 (**612 for NCEP**) Inputs:**
  - *10 Profiles:* temperature; humidity; ozone, methane, cfc11, cfc12, & N_2O mixing ratios, pressure, cloudiness, emissivity
  - *Relevant surface characteristics*: surface pressure, upward LW flux on a surface - *flwupcgs*
- **33 (**69 for NCEP**) Outputs:**
  - Profile of heating rates (26)
  - 7 LW radiation fluxes: *flns, flnt, flut, flnsc, flntc, flutc, flwds*
- **Hidden Layer: One layer with 50 to 300 neurons**
- **Training: *nonlinear optimization in the space with dimensionality of 15,000 to 100,000***
  - Training Data Set: Subset of about 200,000 instantaneous profiles simulated by CAM for the 1-st year
  - Training time: about 1 to several days (SGI workstation)
  - Training iterations: 1,500 to 8,000
- **Validation on Independent Data:**
  - Validation Data Set (independent data): about 200,000 instantaneous profiles simulated by CAM for the 2-nd year

Neural Networks for NCAR (NCEP) SW Radiation

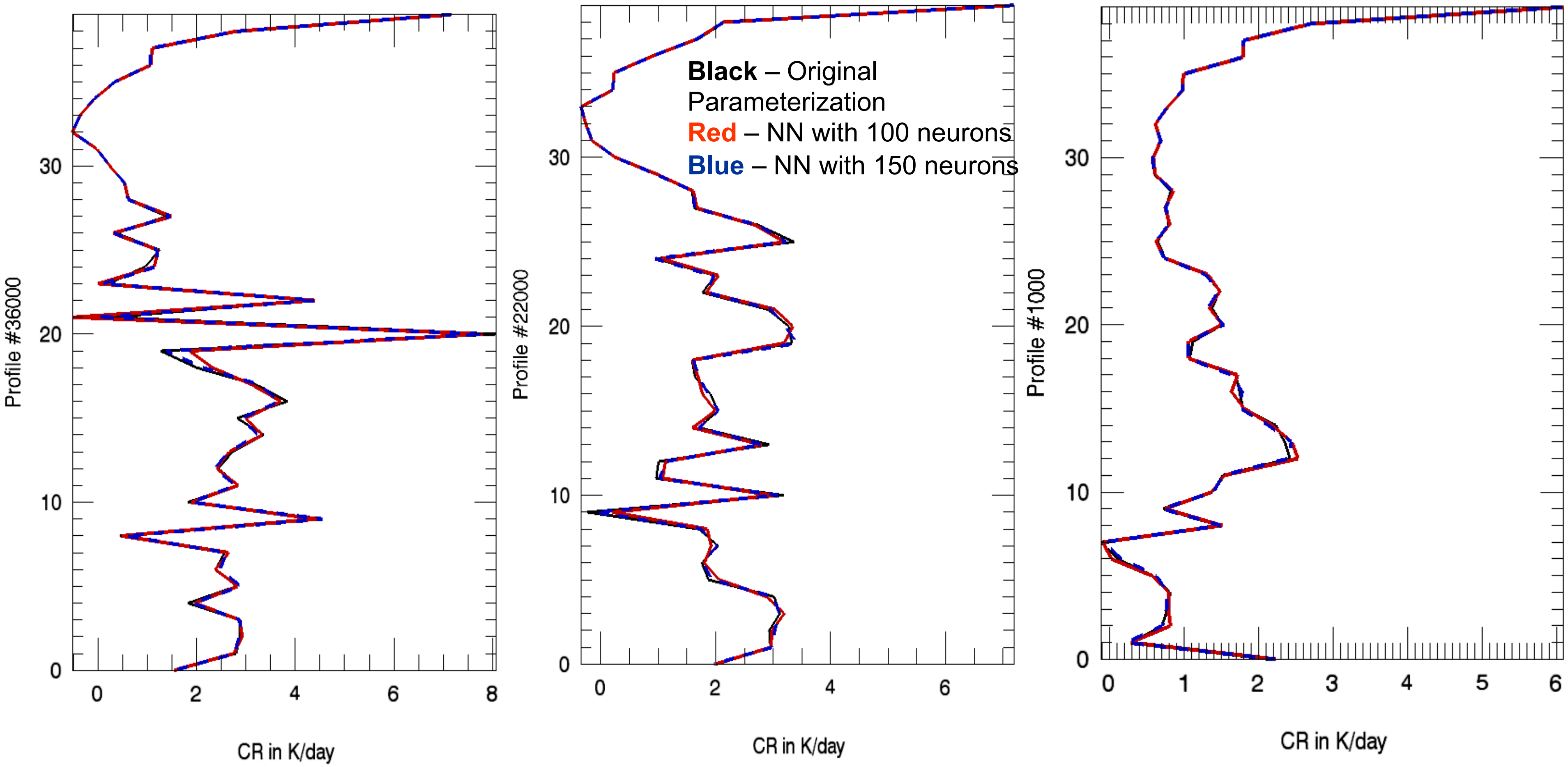
NN characteristics

- **451** (650 NCEP) **Inputs:**
  - ***21 Profiles:*** specific humidity, ozone concentration, pressure, cloudiness, aerosol mass mixing ratios, etc
  - ***7 Relevant surface characteristics***
- **33** (73 NCEP) **Outputs:**
  - Profile of heating rates (26)
  - 7 LW radiation fluxes: *fsns, fsnt, fsdc, sols, soll, solsd, solld*
- **Hidden Layer: One layer with 50 to 200 neurons**
- **Training: *nonlinear optimization in the space with dimensionality of 25,000 to 130,000***
  - Training Data Set: Subset of about 200,000 instantaneous profiles simulated by CAM for the 1-st year
  - Training time: about 1 to several days (SGI workstation)
  - Training iterations: 1,500 to 8,000
- **Validation on Independent Data:**
  - Validation Data Set (independent data): about 100,000 instantaneous profiles simulated by CAM for the 2-nd year

NN Approximation Accuracy and Performance vs. Original Parameterization (*on an independent data set*)

Parameter	Model	Bias	RMSE	Mean	☒	Performance
LWR (☒*K/day*)	**NASA** M-D. Chou	$1.\ 10^{-4}$	0.32	-1.52	1.46	
	NCEP AER *rrtm2*	$7.\ 10^{-5}$	0.40	-1.88	2.28	☒ 100 times faster
	NCAR W.D. Collins	$3.\ 10^{-5}$	0.28	-1.40	1.98	☒ 150 times faster
SWR (☒*K/day*)	**NCAR** W.D. Collins	$6.\ 10^{-4}$	0.19	1.47	1.89	☒ 20 times faster
	NCEP AER *rrtm2*	$1.\ 10^{-3}$	0.21	1.45	1.96	☒ 40 times faster

Individual Profiles



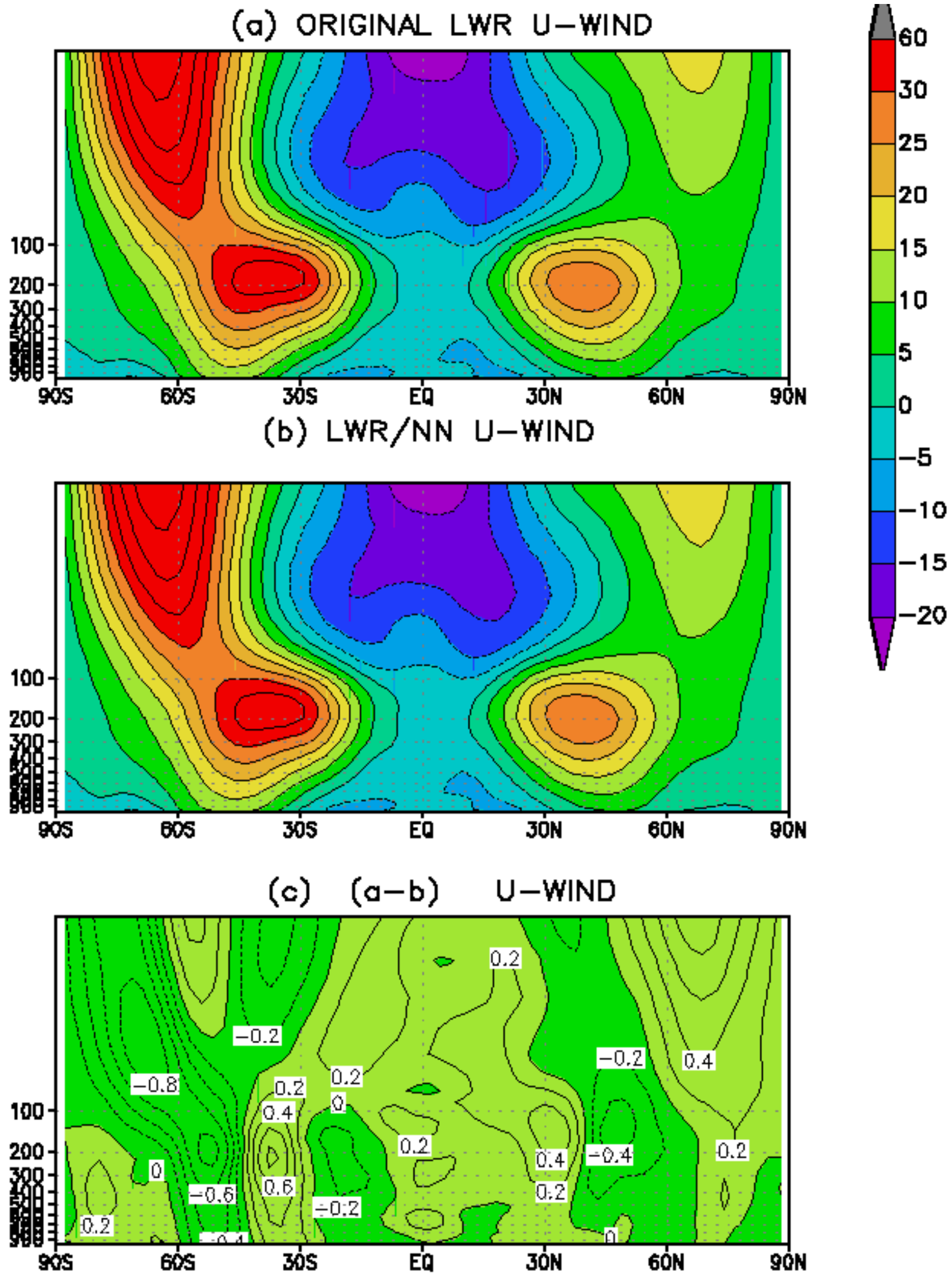
PRMSE = 0.18 & 0.10 K/day **PRMSE = 0.11 & 0.06 K/day** **PRMSE = 0.05 & 0.04 K/day**

NCAR CAM-2: 50 YEAR EXPERIMENTS
NCEP CFS: 17 YEAR EXPERIMENTS

- **CONTROL RUN: the standard NCAR CAM or NCEP CFS versions with the original Radiation (LWR and SWR)**

- **NN RUN: the hybrid version of NCAR CAM or NCEP CFS with NN emulation of the LWR & SWR**

NCAR CAM-2 Zonal Mean U 50 Year Average

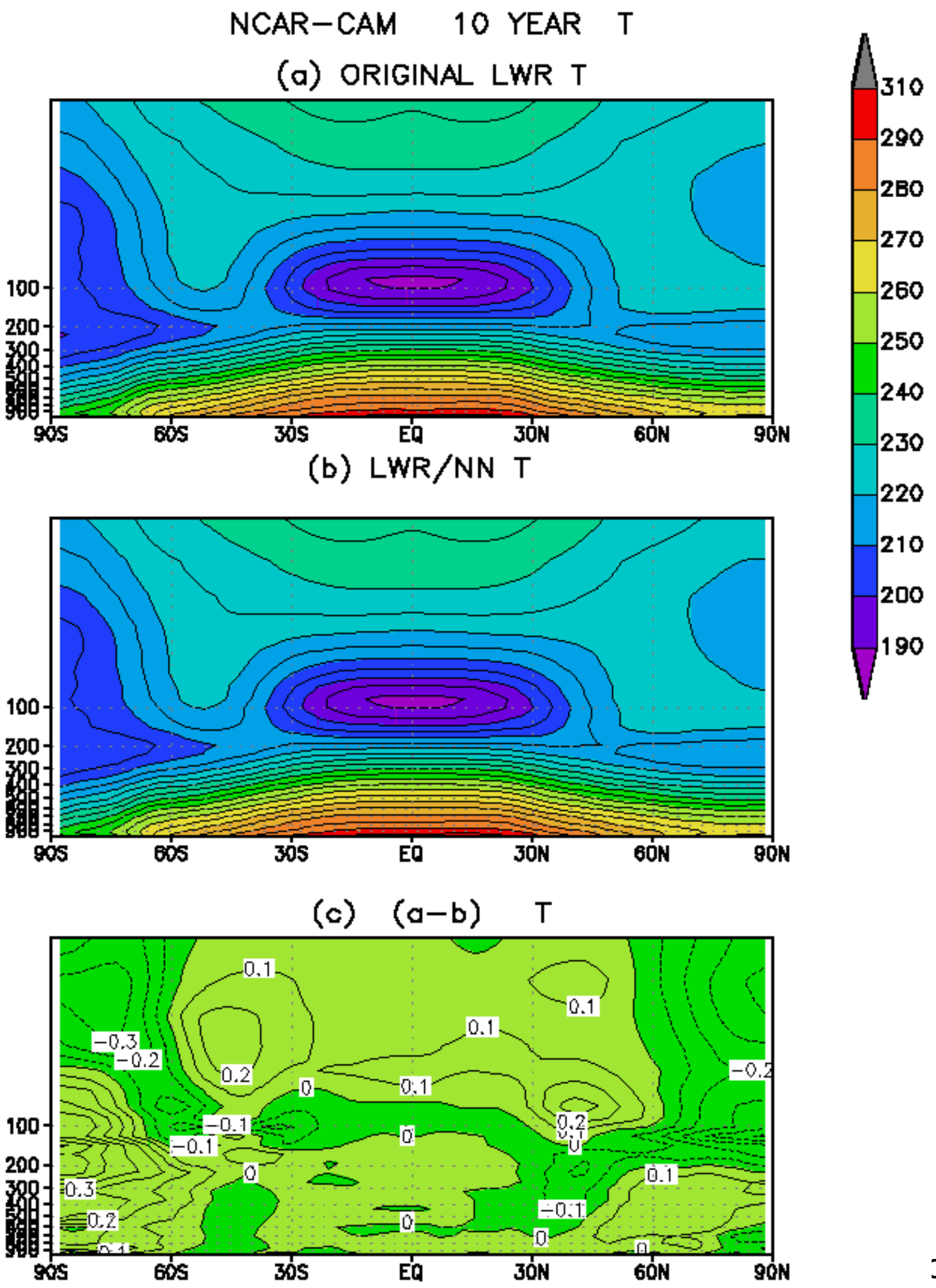


(a)– Original LWR Parameterization

(b)- NN Approximation

(c)- Difference (a) – (b), contour 0.2 *m/sec*

all in *m/sec*



NCAR CAM-2 Zonal Mean Temperature 50 Year Average

(a)– Original LWR Parameterization

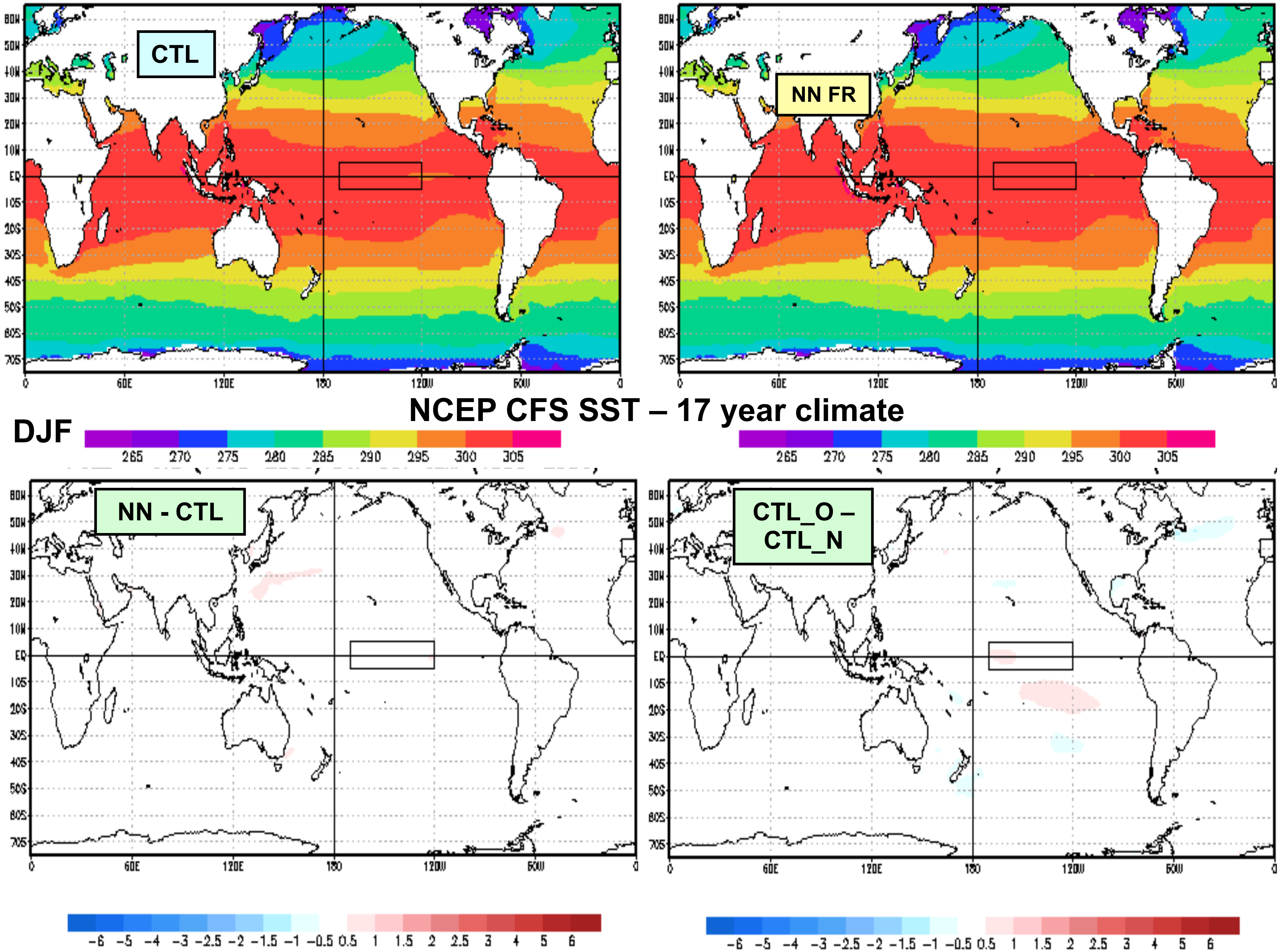
(b)- NN Approximation

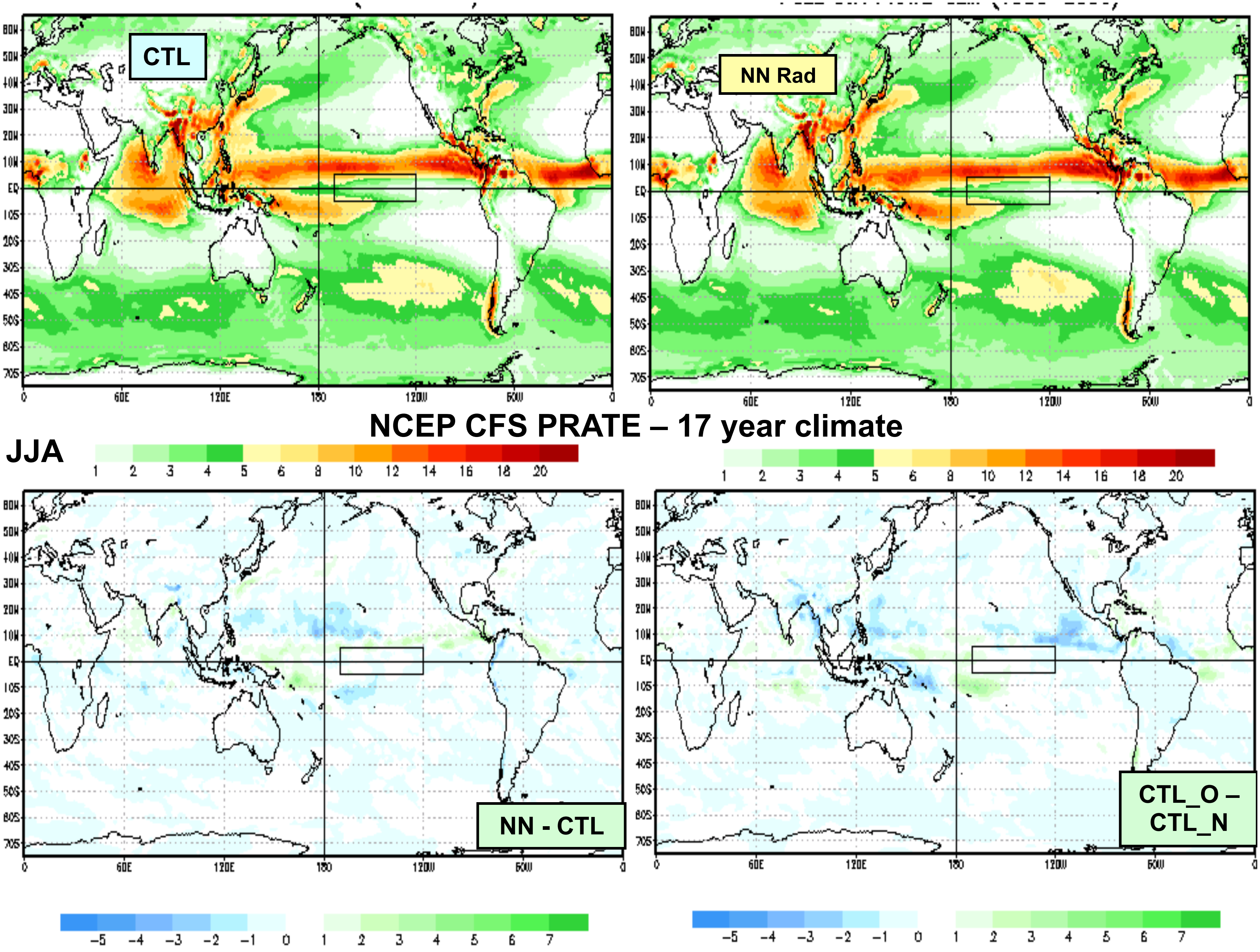
(c)- Difference (a) – (b), contour 0.1 °K

all in °K

NCEP CFS SST – 17 year climate

DJF





NCEP CFS PRATE – 17 year climate

JJA

Application of the Neural Network Technique to Develop a Nonlinear Multi-Model Ensemble for Precipitations over ConUS

Calculating Ensemble Mean

- **Conservative ensemble**

$$EM = 1/N \sum_{i=1}^{N} p_i$$

- **Weighted ensemble**

$$WEM = \sum_{i=1}^{N} W_i p / \sum_{i=1}^{N} W_i$$

W_i from a priori information

or from past data => linear regression

- **If data are available, we can relax assumption of linearity**

$$NEM = f(P) \cong NN(P)$$

Available data for precipitations over ConUS

- Precipitation forecasts available from 8 operational models:
  - NCEP's mesoscale & global models (NAM & GFS)
  - the Canadian Meteorological Center regional & global models (CMC & CMCGLB)
  - global models from the Deutscher Wetterdienst (DWD)
  - the European Centre for Medium-Range Weather Forecasts (ECMWF) global model
  - the Japan Meteorological Agency (JMA) global model
  - the UK Met Office (UKMO) global model
- Also NCEP Climate Prediction Center (CPC) precipitation analysis is available over ConUS.

Data & Products for Comparisons

- Forecasts:
  - **MEDLEY multi-model ensemble: simple average of 8 models (24 hr forecasts)**
  - **NN multi-model ensemble (experimental, 24 hr forecast)**
  - **Hydrometeorological Prediction Center (HPC) human 24 hr forecast, produced by human forecaster using models, satellite images, and other available data**
- Validation: **CPC analysis over ConUS**

MEDLAY

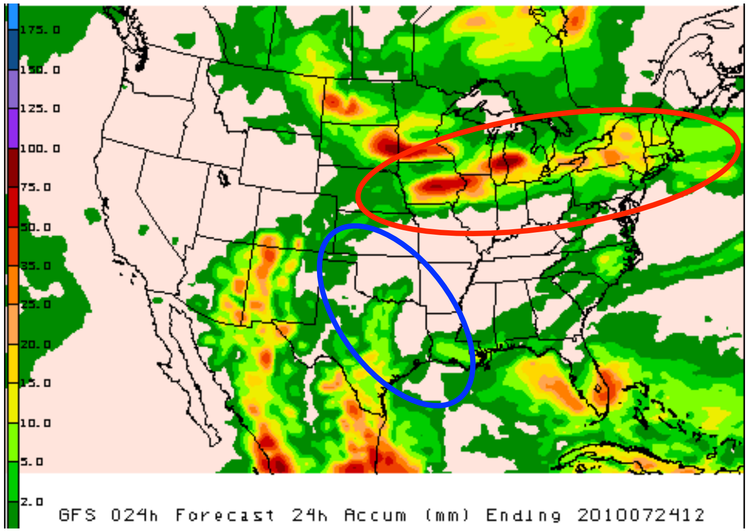
Advantages: **better placement of precipitation areas**

Disadvantages **(because of simple linear averaging)** Motivation for NN developments**:**

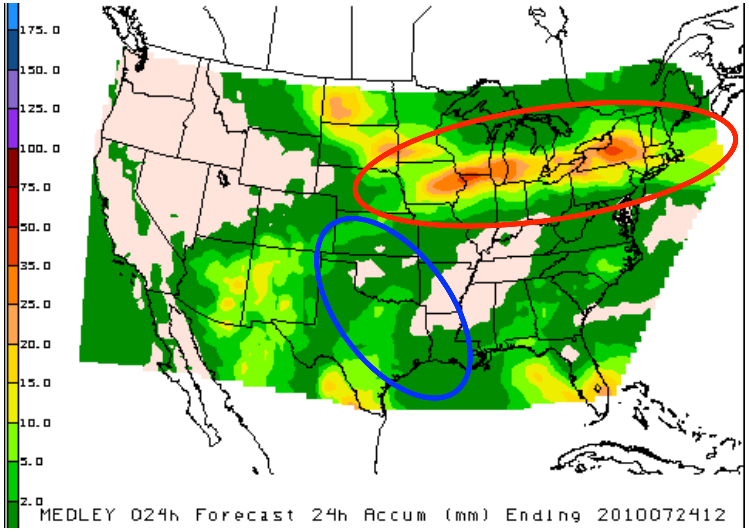
- **Smoothes, diffuse features, reduces gradients**
  - **High bias for low level precip – large areas of false low precip**
  - **Low bias in high level precip – highs smoothed out and reduced**

24h Forecast Ending 07/24/2010 at 12Z

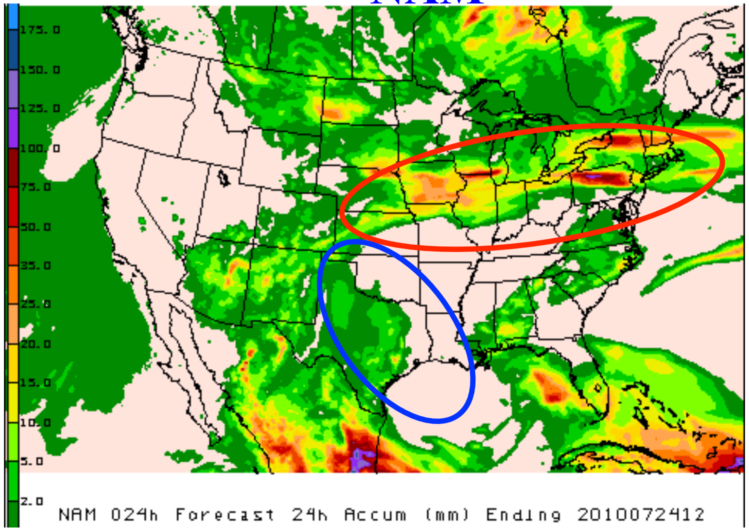
GFS



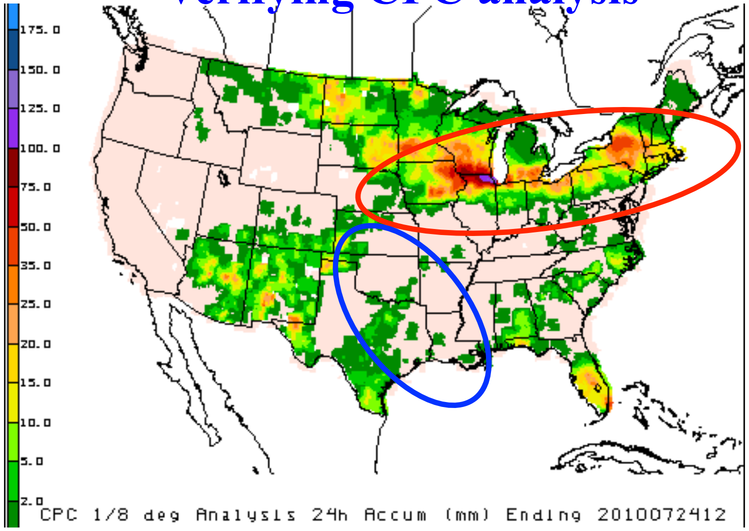
MEDLEY



NAM



Verifying CPC analysis

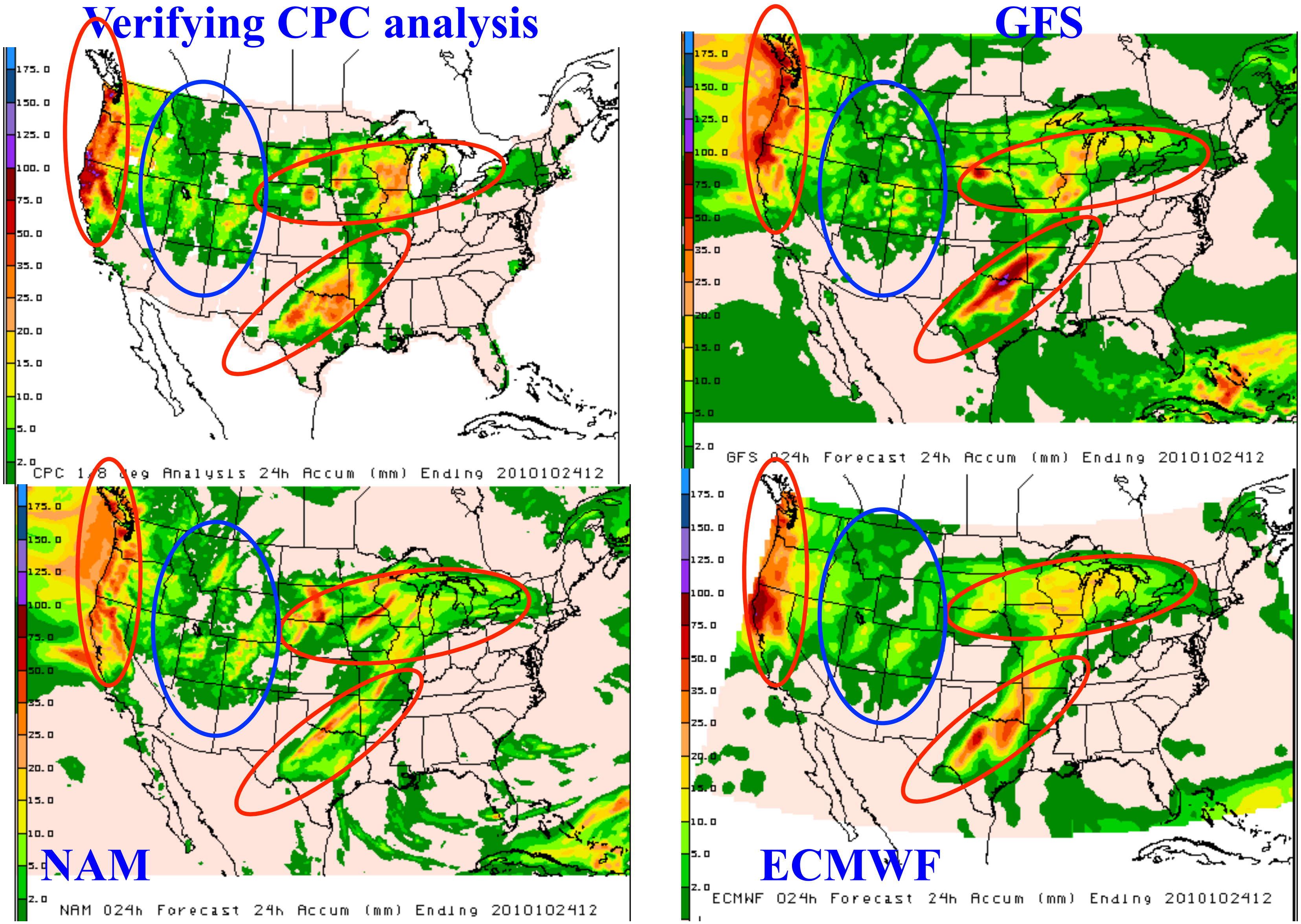


A NN Multi-Model Ensemble

- **Use past data (model forecasts and verifying analysis data) to train NN**
  - **For NN Inputs: precip amounts (8 model 24 hr forecasts), lat, lon, and day of the year**
  - **For NN output: CPC verification analysis for the corresponding time**
- **Data for 2009 have been used for training**

$$NN_{ens} = a_0 + \sum_{j=1}^{k} a_j \cdot \phi(b_{j0} + \sum_{i=1}^{n} b_{ji} \cdot x_i) \qquad ; \;\; \boldsymbol{n = 12; k = 7}$$

Sample NN forecast: example 1 (1)



Verifying CPC analysis

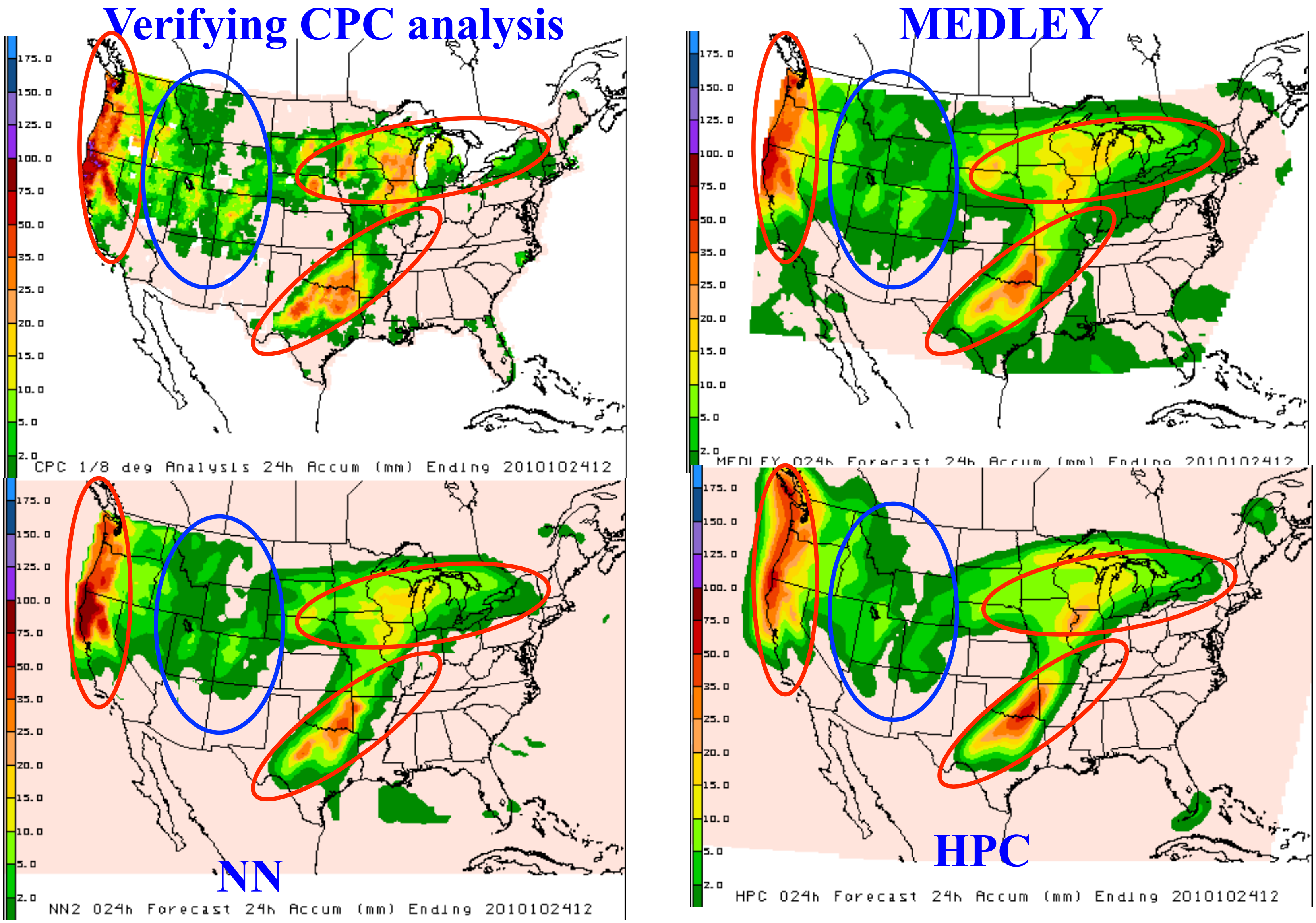
GFS

NAM

ECMWF

175.0
150.0
125.0
100.0
75.0
50.0
35.0
25.0
20.0
15.0
10.0
5.0
2.0
CPC 1/8 deg Analysis 24h Accum (mm) Ending 2010102412
GFS 024h Forecast 24h Accum (mm) Ending 2010102412
NAM 024h Forecast 24h Accum (mm) Ending 2010102412
ECMWF 024h Forecast 24h Accum (mm) Ending 2010102412

Sample NN forecast: example 1 (2)



Verifying CPC analysis

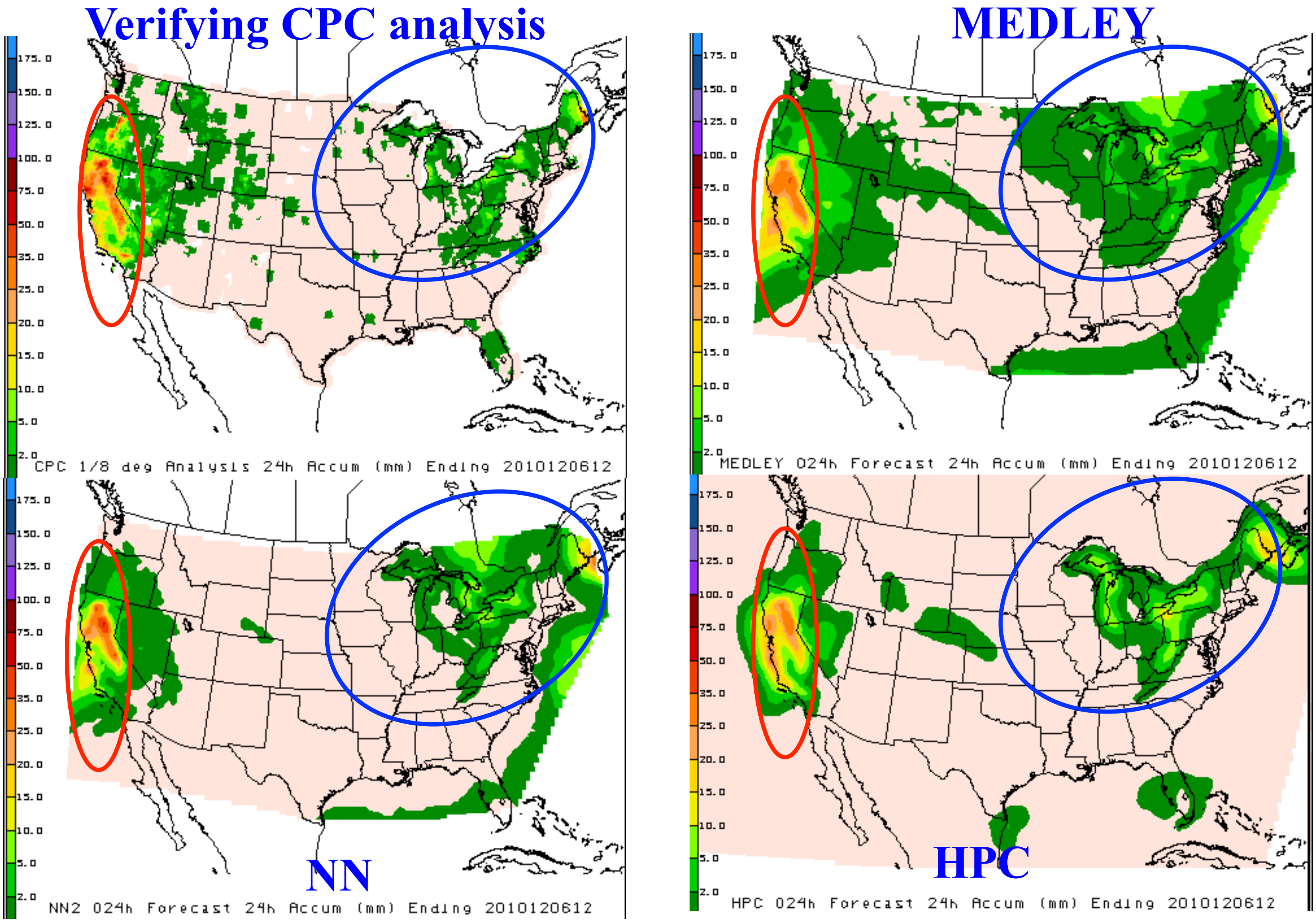
MEDLEY

NN

HPC

175.0
150.0
125.0
100.0
75.0
50.0
35.0
25.0
20.0
15.0
10.0
5.0
2.0
CPC 1/8 deg Analysis 24h Accum (mm) Ending 2010102412
MEDLEY 024h Forecast 24h Accum (mm) Ending 2010102412
NN2 024h Forecast 24h Accum (mm) Ending 2010102412
HPC 024h Forecast 24h Accum (mm) Ending 2010102412

Sample NN forecast: example 2



Verifying CPC analysis

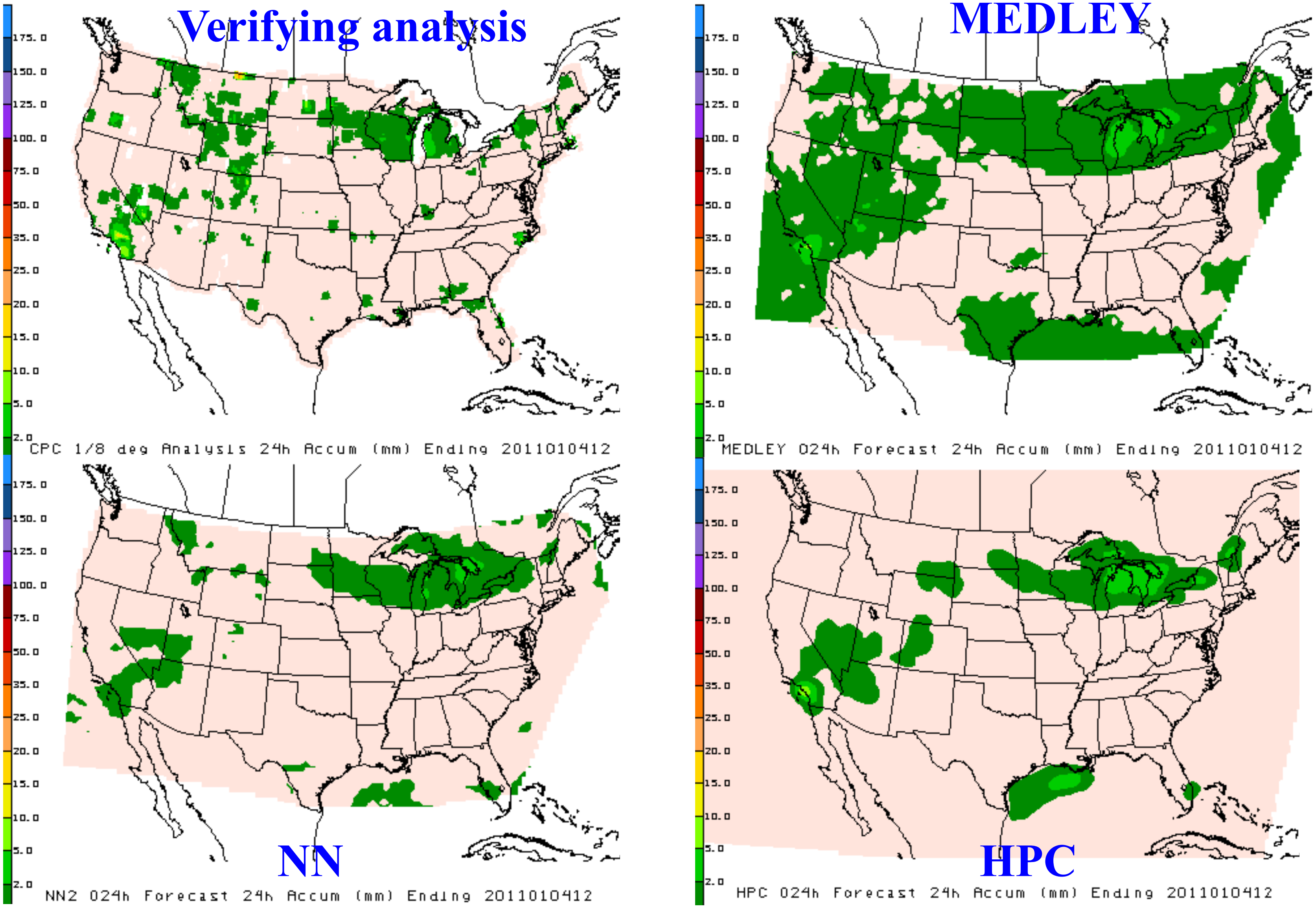
MEDLEY

NN

HPC

175.0
150.0
125.0
100.0
75.0
50.0
35.0
25.0
20.0
15.0
10.0
5.0
2.0
CPC 1/8 deg Analysis 24h Accum (mm) Ending 2010120612
MEDLEY 024h Forecast 24h Accum (mm) Ending 2010120612
NN2 024h Forecast 24h Accum (mm) Ending 2010120612
HPC 024h Forecast 24h Accum (mm) Ending 2010120612

Sample NN forecast: example 3


Verifying analysis
MEDLEY
NN
HPC
175.0
150.0
125.0
100.0
75.0
50.0
35.0
25.0
20.0
15.0
10.0
5.0
2.0
CPC 1/8 deg Analysis 24h Accum (mm) Ending 2011010412
MEDLEY 024h Forecast 24h Accum (mm) Ending 2011010412
NN2 024h Forecast 24h Accum (mm) Ending 2011010412
HPC 024h Forecast 24h Accum (mm) Ending 2011010412

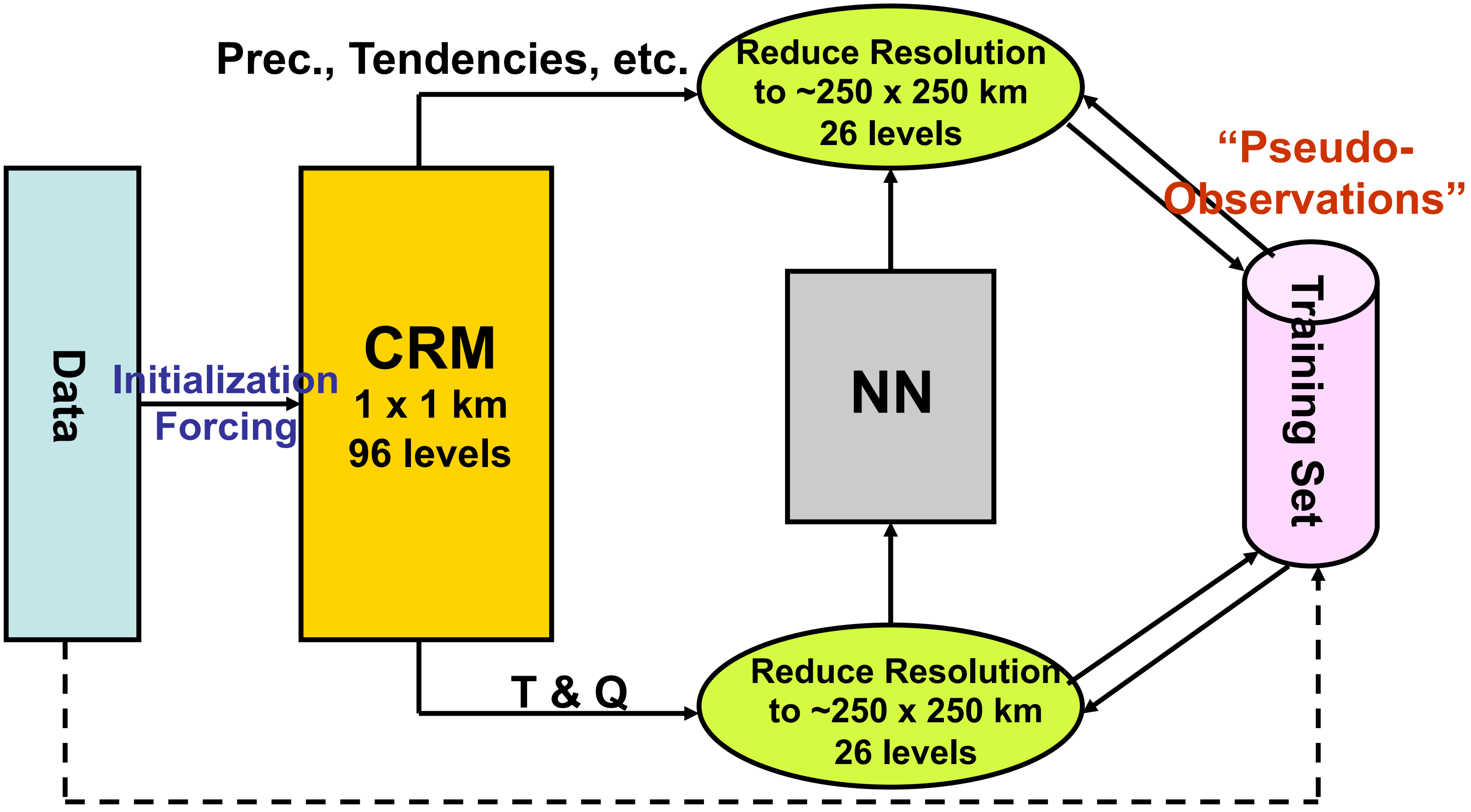
Application of the Neural Network Technique to Develop New NN Convection Parameterization

NN Parameterizations

- New NN parameterizations **of model physics can be developed based on:**
  - **Observations**
  - **Data simulated by first principle process models (like cloud resolving models).**
- **Here NN serves as an interface transferring information about sub-grid scale processes from fine scale data or models (CRM) into GCM (upscaling)**

NN convection parameterizations for climate models based on learning from data.

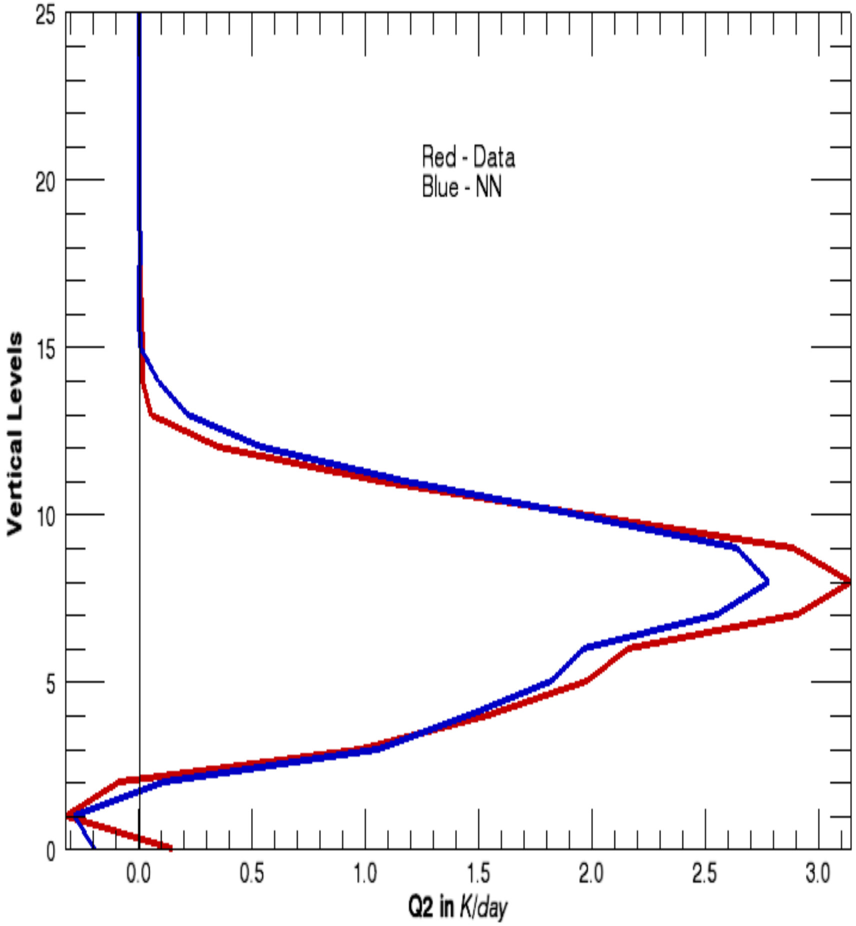
Proof of Concept (POC) -1.



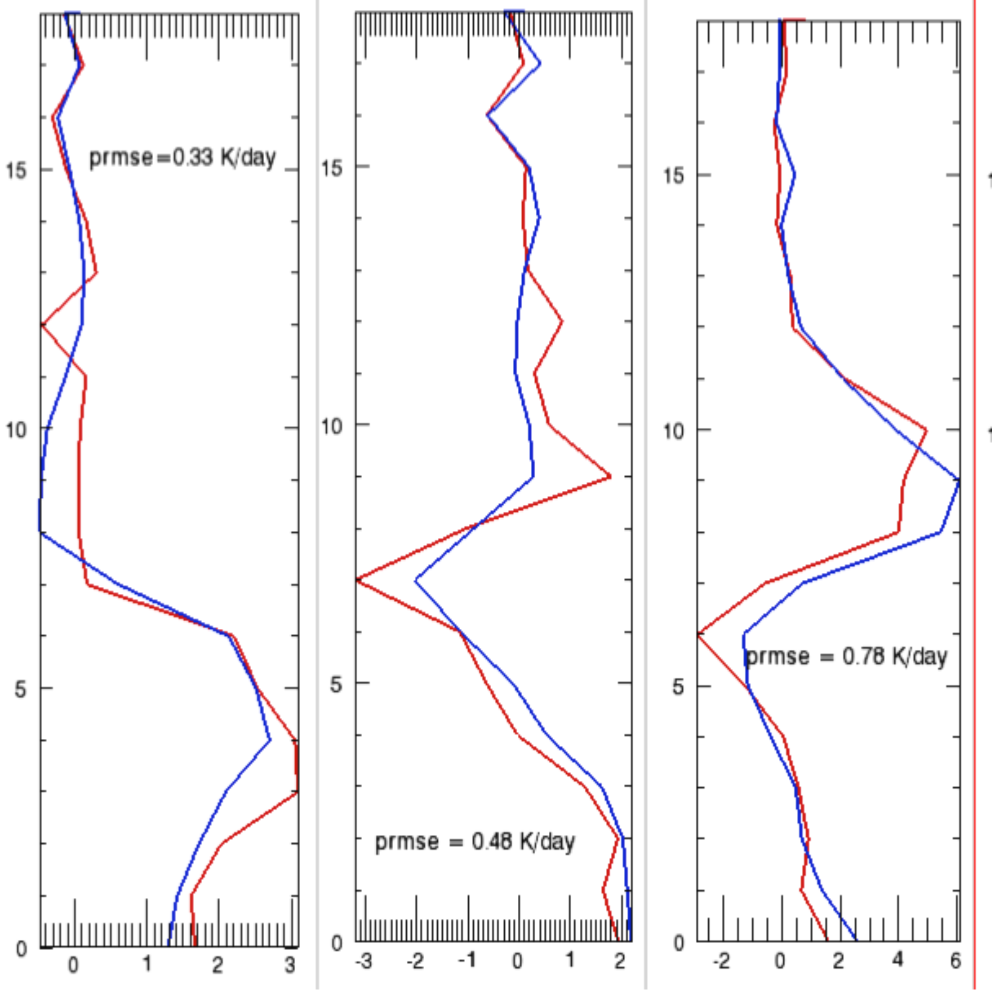
Proof of Concept - 2

- **Data (forcing and initialization): TOGA COARE meteorological conditions**
- **CRM: the SAM CRM (Khairoutdinov and Randall, 2003).**
  - **Data from the archive provided by C. Bretherton and P. Rasch (*Blossey et al, 2006*).**
  - **Hourly data over 90 days**
  - **Resolution 1 km over the domain of 256 x 256 km**
  - **96 vertical layers (0 – 28 km)**
- Resolution of "pseudo-observations" (averaged CRM data)**:**
  - **Horizontal 256 x 256 km**
  - **26 vertical layers**
- NN inputs: only **temperature and water vapor fields; a limited training data set used for POC**
- NN outputs: **precipitation & the tendencies T and q, i.e. "apparent heat source" (Q1), "apparent moist sink" (Q2), and cloud fractions (CLD)**

Proof of Concept - 4

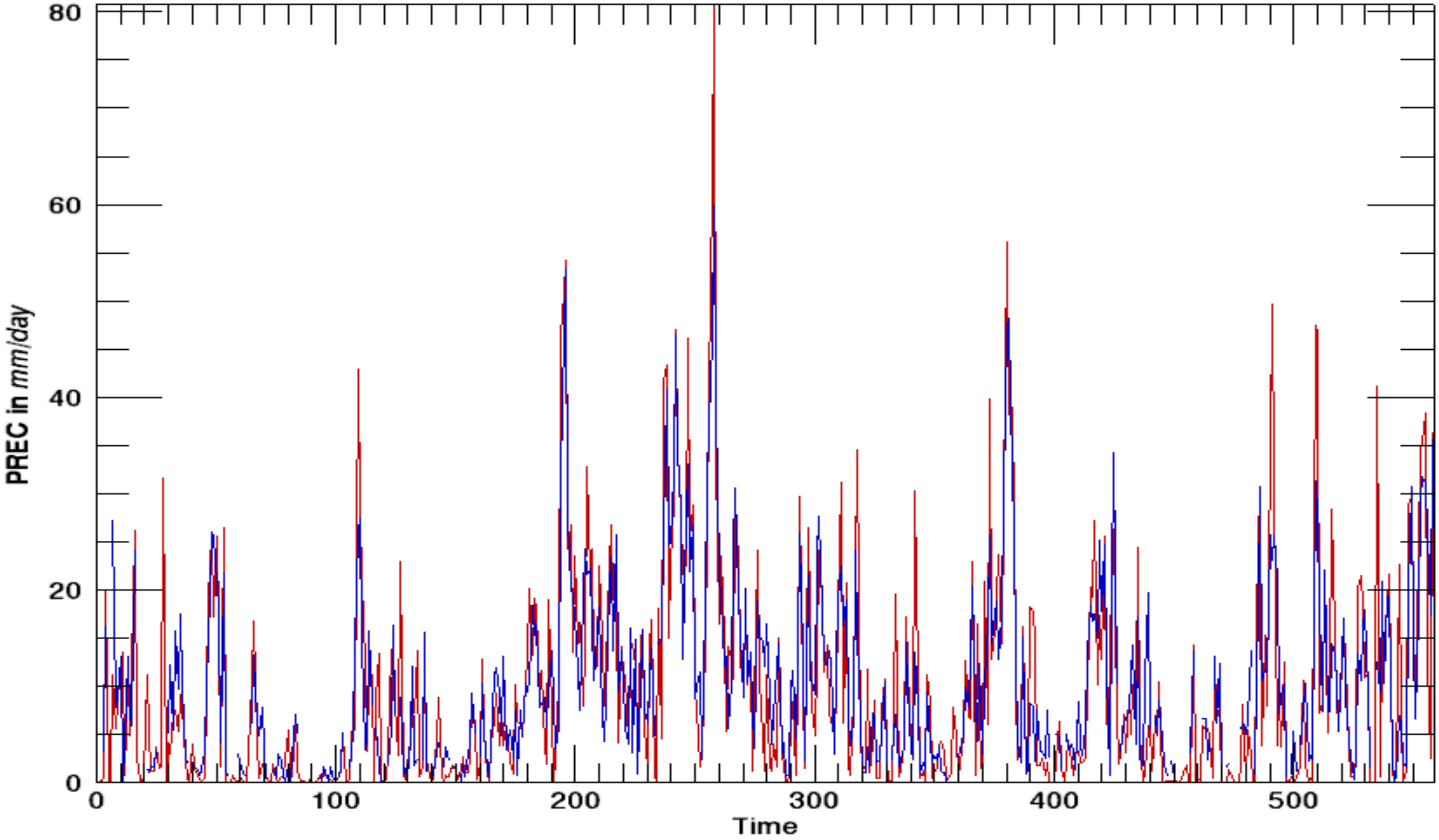


Time averaged water vapor tendency (expressed as the equivalent heating) for the validation dataset.



Q2 profiles (red) with the corresponding NN generated profiles (blue). The profile rmse increases from the left to the right.

Proof of Concept - 3

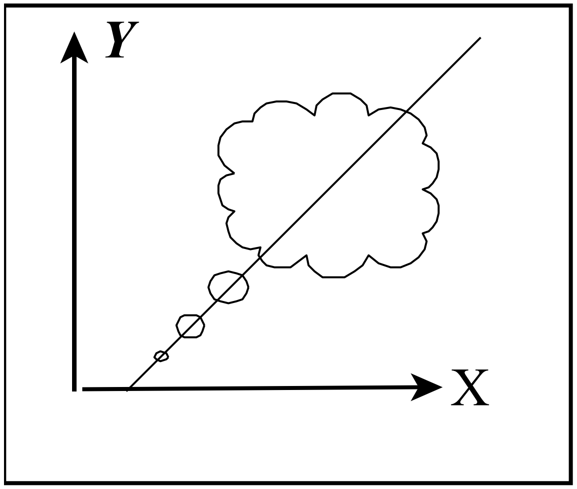


Precipitation rates for the validation dataset. Red – data, blue - NN

How to Develop NNs:
An Outline of the Approach (1)

- **Problem Analysis:**
  - **Are traditional approaches unable to solve your problem?**
    - **At all**
    - **With desired accuracy**
    - **With desired speed, etc.**
  - **Are NNs well-suited for solving your problem?**
    - **Nonlinear mapping**
    - **Classification**
    - **Clusterization, etc.**
  - **Do you have a first guess for NN architecture?**
    - **Number of inputs and outputs**
    - **Number of hidden neurons**

How to Develop NNs:
An Outline of the Approach (2)



- Data Analysis
  - How noisy are your data?
    - May change architecture or even technique
  - Do you have enough data?
  - For selected architecture:
    - 1) Statistics => $N^1{}_A > n_W$
    - 2) Geometry => $N^2{}_A > 2^n$
    - $N^1{}_A < N_A < N^2{}_A$
    - To represent all possible patterns => N_R
$N_{TR} = \max(N_A, N_R)$
  - Add for test set: $N = N_{TR} \times (1 + \tau); \tau > 0.5$
  - Add for validation: $N = N_{TR} \times (1 + \tau + \nu); \nu > 0.5$

How to Develop NNs:
An Outline of the Approach (3)

- Training
  - Try different initializations
  - If results are not satisfactory, then goto Data Analysis or Problem Analysis
- Validation (must for any nonlinear tool!)
  - Apply trained NN to independent validation data
  - If statistics are not consistent with those for training and test sets, go back to Training or Data Analysis

Conclusions

- There is an obvious trend in scientific studies:
  - From simple, linear, single-disciplinary, low dimensional systems
  - To complex, nonlinear, multi-disciplinary, high dimensional systems
- There is a corresponding trend in math & statistical tools:
  - From simple, linear, single-disciplinary, low dimensional tools and models
  - To complex, nonlinear, multi-disciplinary, high dimensional tools and models
- Complex, nonlinear tools have advantages & limitations: learn how to use advantages & avoid limitations!
- Check your toolbox and follow the trend, otherwise you may miss the train!

Recommended Reading

- **Regression Models:**
  - B. Ostle and L.C. Malone, “Statistics in Research”, 1988
- **NNs, Introduction:**
  - R. Beale and T. Jackson, “Neural Computing: An Introduction”, 240 pp., Adam Hilger, Bristol, Philadelphia and New York., 1990
- **NNs, Advanced:**
  - Bishop Ch. M., 2006: Pattern Recognition and Machine Learning, Springer.
  - V. Cherkassky and F. Muller, 2007: Learning from Data: Concepts, Theory, and Methods, J. Wiley and Sons, Inc
  - Haykin, S. (1994), *Neural Networks: A Comprehensive Foundation*, 696 pp., Macmillan College Publishing Company, New York, U.S.A.
  - Ripley, B.D. (1996), *Pattern Recognition and Neural Networks*, 403 pp., Cambridge University Press, Cambridge, U.K.
  - Vapnik, V.N., and S. Kotz (2006), *Estimation of Dependences Based on Empirical Data (Information Science and Statistics)*, 495 pp., Springer, New York.
- **NNs in Environmental Sciences:**
  - Krasnopolsky, V., 2007: “Neural Network Emulations for Complex Multidimensional Geophysical Mappings: Applications of Neural Network Techniques to Atmospheric and Oceanic Satellite Retrievals and Numerical Modeling”, *Reviews of Geophysics*, 45, RG3009, doi: 10.1029/2006RG000200.
  - Hsieh, W., 2009: “Machine Learning Methods in the Environmental Sciences”, Cambridge University Press, 349 pp.