ENGO 697

Remote Sensing Systems and Advanced Analytics

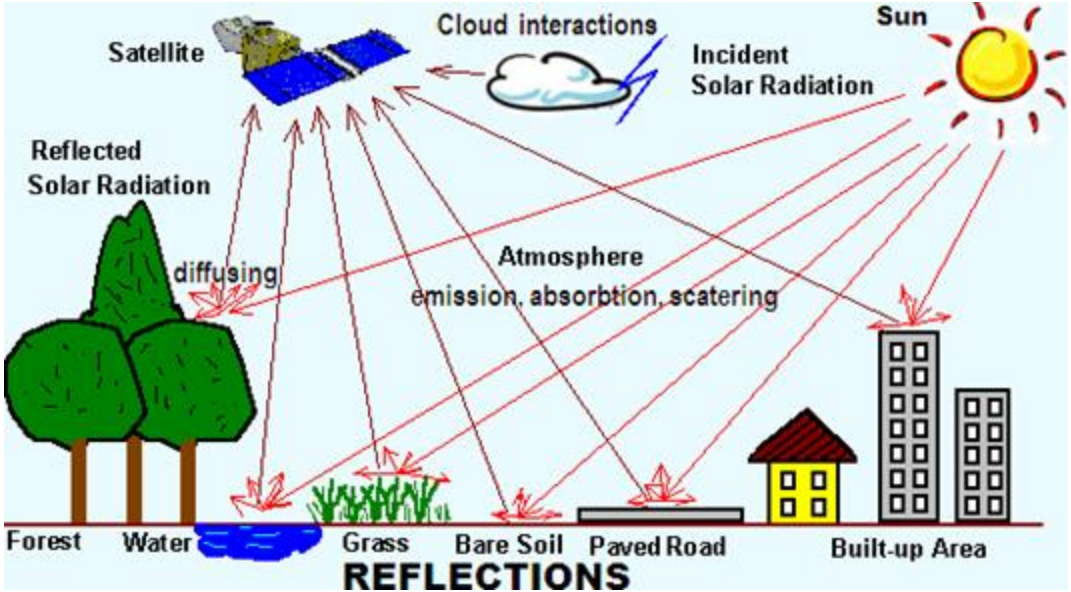
Session 7: How to solve inverse problems in remote sensing systems

Dr. Linlin (Lincoln) Xu
Linlin.xu@ucalgary.ca
Office: ENE 221

Outline

- ➔ Overview
- ➔ Direct inversion
- ➔ Numerical approach
- ➔ Look up table (LUT)
- ➔ Simulation & Machine learning (ML)
- ➔ Data -driven ML and DL
- ➔ Bayesian framework
- ➔ Questions

Remote Sensing System Overview



Forward model:

$Y = f(X)$

(1) Y: radiation received by the sensor

(2) X: variables that you want to estimate, e.g., class labels, chlorophyll content in leaves, leaf area index/density;

Inverse model:

$X = g(Y, \theta)$

where g(.) is an unknown inverse function with unknown model parameter θ.

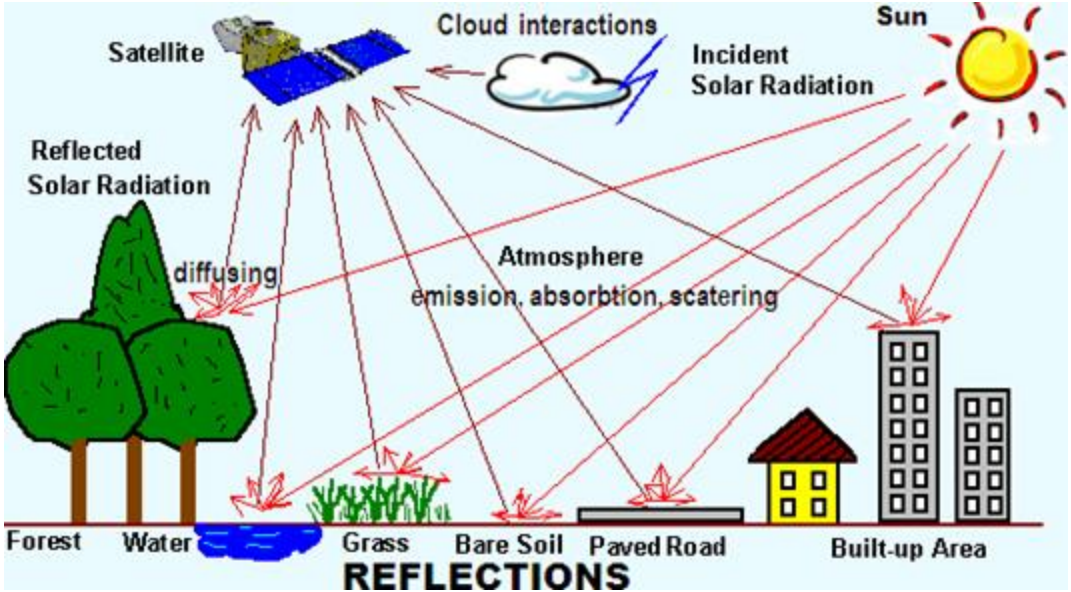
Why estimating X is difficult?

--- Knowledge f(.) complex, biased, highly nonlinear, with large uncertainty;

---- Data (X, Y) pairs limited, poor quality;

---- Prior information (e.g., spatial prior) ambiguous;

Remote Sensing System Overview



Forward model: $Y = f(X)$

Inverse model: $X = g(Y, \theta)$

What are the approaches for solving the inverse problems? Advantages? Disadvantages?

Common approaches for solving inverse model:

1. If f(.) is known and invertible, use direct inverse function approach.

1. If f(.) is known but highly nonlinear,
   2.1 use numerical approaches, e.g., Newton's approach, to estimate X using observations {Y}.
   2.2 use the lookup table (LUT) approach to simulate {(X,Y)} pairs for estimating {X} of observations {Y}.
   2.3 simulate {(X,Y)} pairs to train inverse function g(.) which will be tested on real observations {Y}.

1. If f(.) is unknown, requires (Y, X) pairs, use ML and DL, to empirically build g(.).

(1) Direct Inverse Function Approach

In an ideal scenario, the forward model f(.) is simple and invertible, we can use direct inverse function approach to estimate X.

Forward model: $Y = f(X)$

where f(.) is invertible, f(.) -> $f^{-1}(.)$. For example,

Forward model: $Y = X^2$

Inverse function: $X = Y^{0.5}$

Given Y, X can be estimated based on the above inverse function.

However, in real-world remote sensing systems, the forward model f(.) are complicated radiative transfer models, which are usually highly nonlinear and non-invertible, and as such they do not have inverse functions.

Questions: Can this approach use data (X,Y) pairs, and spatial prior information?

Planck's Law

$$B(\lambda,T) = c_1 / \lambda^5 / [e^{c_2/\lambda T} - 1] \quad (W/m^2/ster/um)$$

where λ = wavelengths in um

T = temperature of emitting surface (deg K)

$c_1 = 1.191044 \times 10^{-5}$ ($mW/m^2/ster/cm^{-4}$)

$c_2 = 1.438769$ (cm deg K)

Brightness Temperature

$$T = c_2 / [\lambda \ln(\frac{c_1}{\lambda^5 B_\lambda} + 1)]$$ is determined by inverting Planck function.

	(1) Direct inversion	(2) LUT approach	(3) Numerical Approach	(4) Simulation & ML	(5) ML	(6) DL
f(.) is known	**yes**	yes	yes	yes	yes	yes
f(.) is partially known, i.e., form known, but with some unknown parameters U	**no**	no	Yes, estimate X and U together	no	no	no
f(.) unknown, (X,Y) known	**no**	no	no	no	yes	yes
f(.) unknown, (X,Y) unknown	**no**	no	no	no	no	no
If both f(.) and (X,Y) known, can accommodate both?	**no**	yes?	Yes? Use (X,Y) to estimate parameters in f(.)	yes?	Yes, use both simulated and observed data	Yes, use both simulated and observed data
Can use prior information? e.g., spatial prior and value prior	**no**	Yes? Use value prior for sampling	Yes? Use value prior of X in Bayesian estimation	Yes, Use value prior in sampling and spatial prior in Random fields	Yes, spatial prior in Random field approaches	Yes, similar to ML
Advantages	**Knowledge-driven; Simple, easy**	Knowledge-driven; Intuitive, easy, discrete fitting;	Knowledge-driven; estimate U; Efficient for simple f(.) in convex problems	Knowledge-driven; flexible; continuous fitting; good inter/extrapolation; faster than LUT	Data-driven; flexible; Classic;	Strong modeling capability; automatic feature learning;
Disadvantages	**Unrealistic; rely on simple f(.)**	Sensitive to accuracy of f(.), similarity metrics, sampling density and range; slow if LUT is large; bad for extrapolation;	Rely on efficiency of nonlinear solver; Slow; Local optimum;	Overfitting and underfitting risk to simulated data; difficult model selection; Sensitive to accuracy of f(.), similarity metrics, sampling density and range;	Weak modeling capability; Rely on “good” engineered features; Black-box; Overfitting, underfitting; Feature and model selection is difficult and slow	Overfitting and underfitting; Black-box;

(2) Numerical Approaches

If the radiative transfer model f(.) is known, and we have an remote sensing observation Y, we can use the numerical approach to estimate the associated X.

Forward model: Y = f(X), where f(.) is the radiative transfer models, which tend to be highly nonlinear and un-invertible.

Based on some observations {Y}, we can build an objective function:

J(X) = Y-f(X)

X = min J(X)

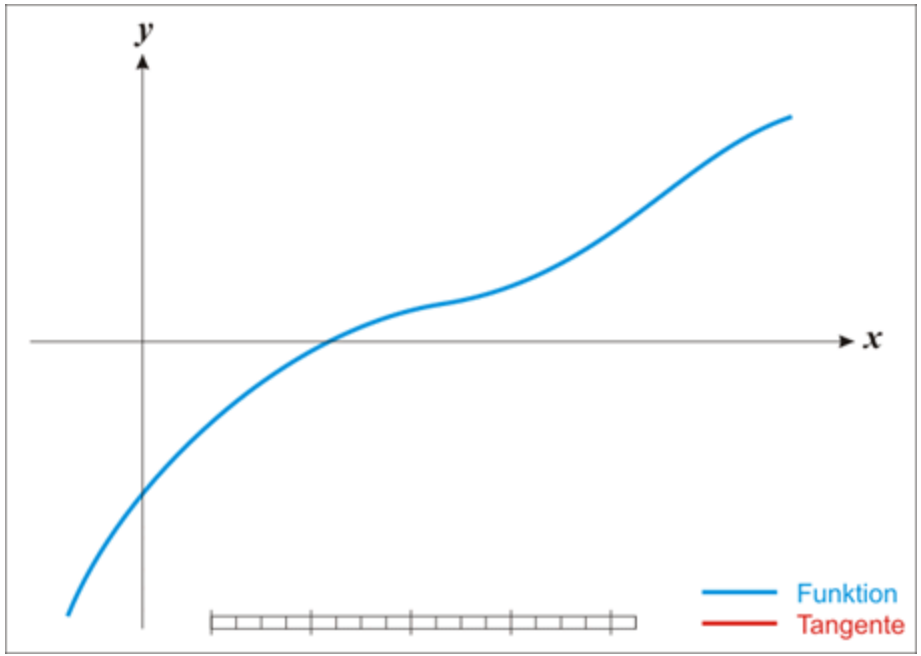
We try to find X that through f(.) can generate output whose value is very close to the value of Y.

There are many methods that can solve this nonlinear optimization problems, for example,
---- Newton's method
---- Gradient descent methods
---- Simulated annealing approach

Because the forward model f(.) contains knowledge and physical rules, this approach is usually called physical model.

Questions: How to find X that can minimize J(X)? Try different X and see which one gives you smallest J(X)? Better approach?

Root-finding Algorithm: Newton's Method



From wikipedia.org

Start with initial guess, and iteratively improve it using its tangent.

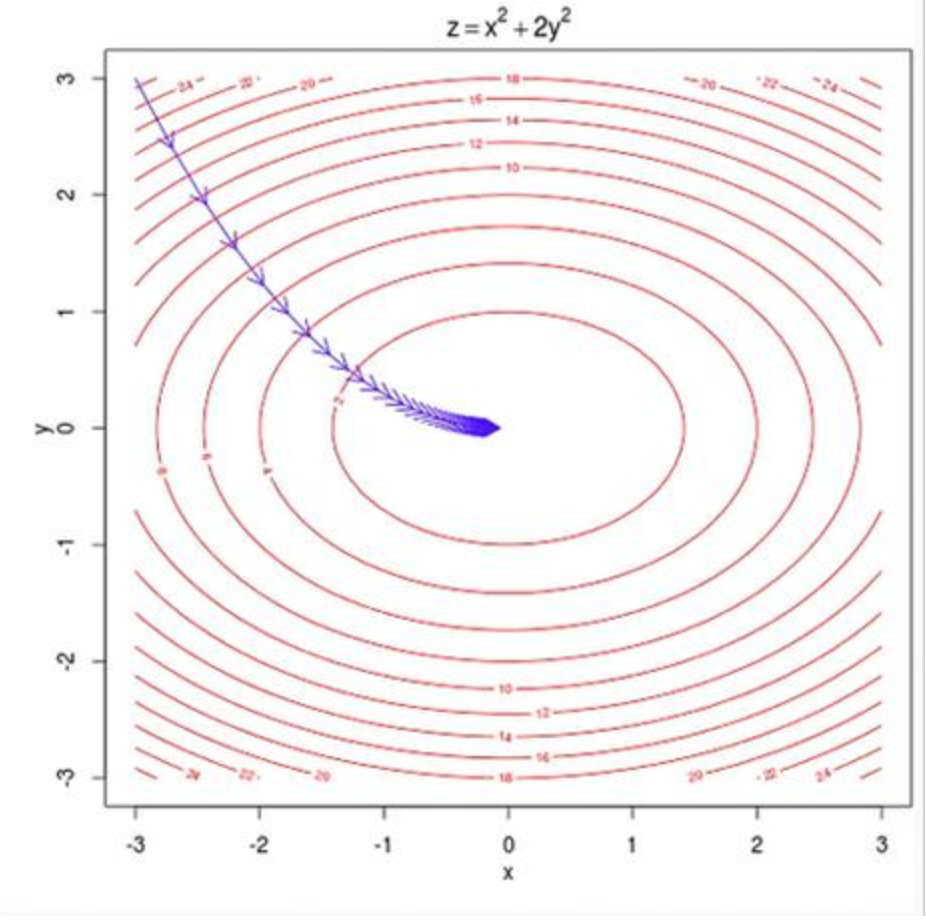
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

The function *f* is shown in blue and the tangent line is in red.

We see that x_{n+1} is a better approximation than x_n for the root x of the function f.

The assumption is the *x*-intercept will typically be a better approximation to the original function's root than the previous guess, and the method can be iterated.

Gradient Descent



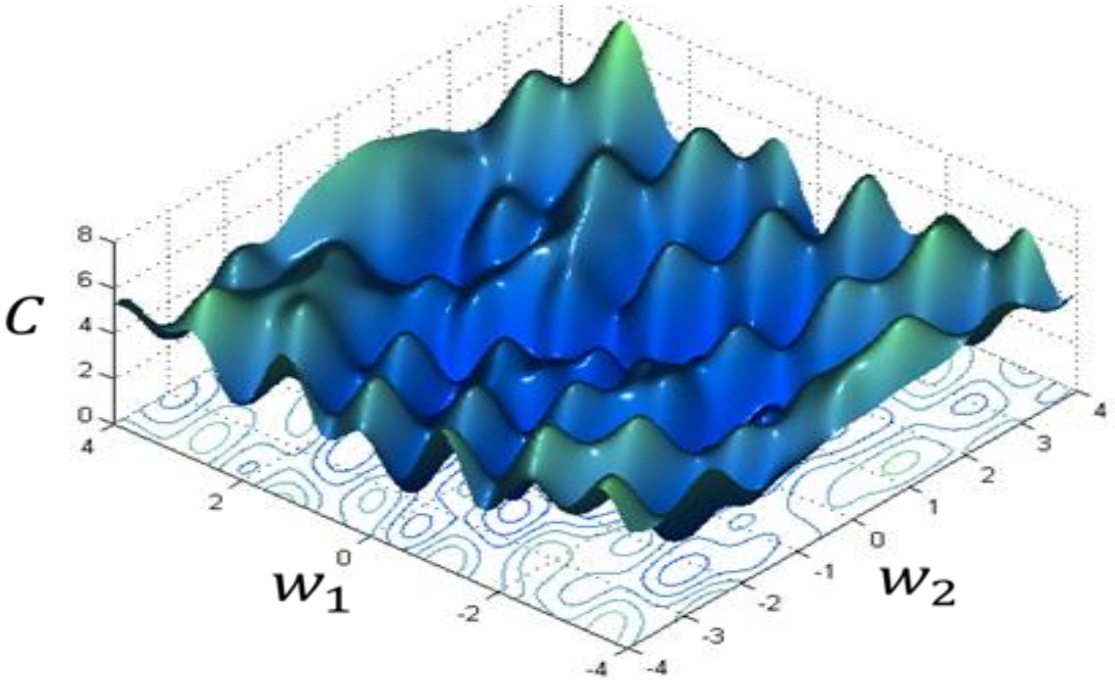
Q1: What if there are multiple local minimums for complex f(.)?

Q2: What are roles of starting point and step length?

Start with initial guess, and iteratively improve it using its gradient.

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \gamma \nabla F(\mathbf{a}_n)$$

Local Minima



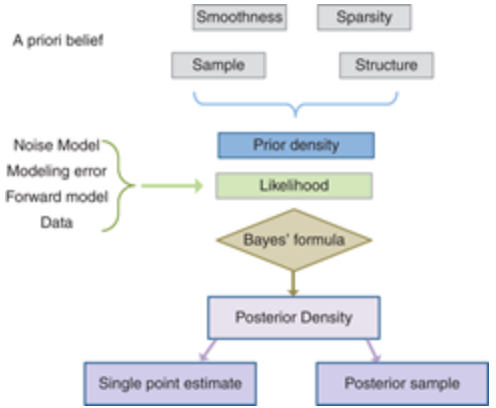
From https://ml4a.github.io

A more realistic scenario where there are many local minima;

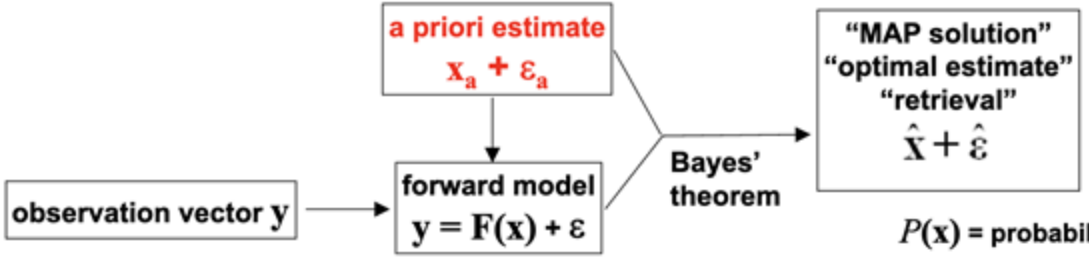
Different initializations lead to different local minima;

How do you find global minima?

Bayesian framework - how to use "prior" information and accommodate errors?



Optimize values of an ensemble of variables (*state vector* x) using observations:

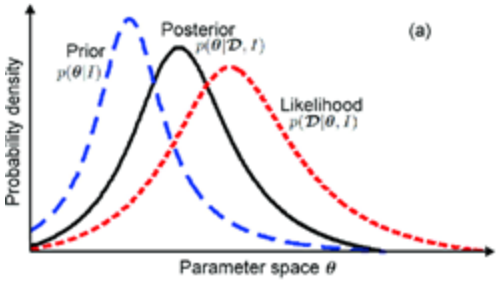


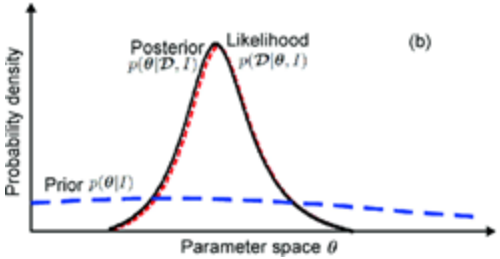
$P(\mathbf{x})$ = probability distribution function (pdf) of $\mathbf{x}$

$P(\mathbf{x},\mathbf{y})$ = pdf of $(\mathbf{x},\mathbf{y})$

$P(\mathbf{y}|\mathbf{x})$ = pdf of $\mathbf{y}$ given $\mathbf{x}$

$$P(\mathbf{x},\mathbf{y})d\mathbf{x}d\mathbf{y} \begin{cases} = P(\mathbf{x})d\mathbf{x}P(\mathbf{y}\,|\,\mathbf{x})d\mathbf{y} \\ = P(\mathbf{y})d\mathbf{y}P(\mathbf{x}\,|\,\mathbf{y})d\mathbf{x} \end{cases}$$





observation pdf *a priori* pdf

a posteriori pdf

$$\Rightarrow \quad P(\mathbf{x}\,|\,\mathbf{y}) = \frac{P(\mathbf{y}\,|\,\mathbf{x})P(\mathbf{x})}{P(\mathbf{y})}$$

Bayes' theorem

normalizing factor (unimportant)

Maximum *a posteriori* (MAP) solution for x given y is defined by $\max(P(\mathbf{x}\,|\,\mathbf{y}))$

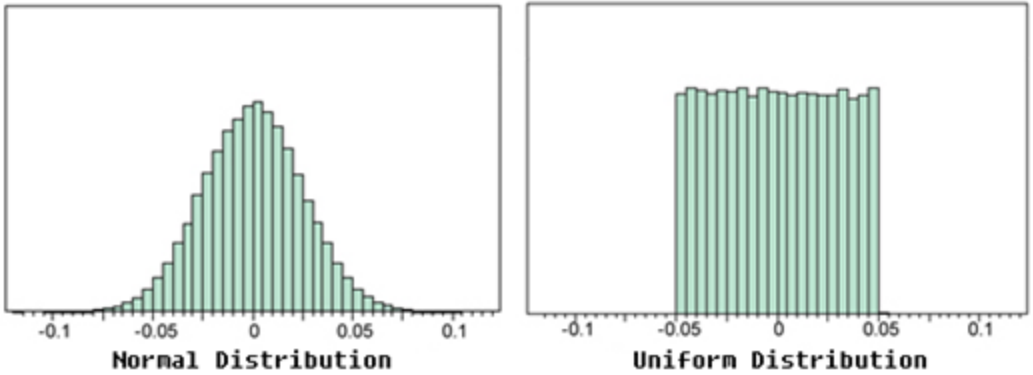
⇨ **solve for** $\nabla_{\mathbf{x}} P(\mathbf{x}\,|\,\mathbf{y}) = \mathbf{0}$

	(1) Direct inversion	(2) LUT approach	**(3) Numerical Approach**	(4) Simulation & ML	(5) ML	(6) DL
f(.) is known	yes	yes	**yes**	yes	yes	yes
f(.) is partially known, i.e., form known, but with some unknown parameters U	no	no	**Yes, estimate X and U together**	no	no	no
f(.) unknown, (X,Y) known	no	no	**no**	no	yes	yes
f(.) unknown, (X,Y) unknown	no	no	**no**	no	no	no
If both f(.) and (X,Y) known, can accommodate both?	no	yes?	**Yes? Use (X,Y) to estimate parameters in f(.)**	yes?	Yes, use both simulated and observed data	Yes, use both simulated and observed data
Can use prior information? e.g., spatial prior and value prior	no	Yes? Use value prior for sampling	**Yes? Use value prior of X in Bayesian estimation**	Yes, Use value prior in sampling and spatial prior in Random fields	Yes, spatial prior in Random field approaches	Yes, similar to ML
Advantages	Knowledge -driven; Simple, easy	Knowledge-driven; Intuitive, easy, discrete fitting;	**Knowledge-driven; estimate U; Efficient for simple f(.) in convex problems**	Knowledge-driven; flexible; continuous fitting; good inter/extrapolation; faster than LUT	Data-driven; flexible; Classic;	Strong modeling capability; automatic feature learning;
Disadvantages	Unrealistic; rely on simple f(.)	Sensitive to accuracy of f(.), similarity metrics, sampling density and range; slow if LUT is large; bad for extrapolation;	**Rely on efficiency of nonlinear solver; Slow; Local optimum;**	Overfitting and underfitting risk to simulated data; difficult model selection; Sensitive to accuracy of f(.), similarity metrics, sampling density and range;	Weak modeling capability; Rely on "good" engineered features; Black-box; Overfitting, underfitting; Feature and model selection is difficult and slow	Overfitting and underfitting; Black-box;

Prosail Simulation - how to simulate data?

TABLE I: Ranges of the input variables for the PROSAIL model for the generation of the LUT.

Variable	Abbr.	Unit	Min	Max
Leaf structure parameter	N	Unitless	1	2
Leaf chlorophyll concentration	C_{ab}	$\mu g cm^{-2}$	20	70
Dry matter content	C_m	gcm^{-2}	0.004	0.007
Equivalent water thickness	C_w	gcm^{-2}	0.005	0.03
Leaf area index	LAI	$m^2 cm^{-2}$	0.001	6
Average leaf angle	ALA	Deg	30	70
Hot-spot size parameter	hot	mm^{-1}	0.05	1
Soil brightness parameter	$scale$	Unitless	0.5	1.5



Prosail is a forward model:

Y = f(X)

What is Y?

1. Bidirectional reflectance from canopy (400nm - 2500nm);

What are the factors that constitute X:

1. A total of 14 input parameters;

How to simulate Y using X?

Step 1: know the distribution of X;

Step 2: obtain samples {Xi| i=1,..,N} based on the distribution of X;

Step 3: use {Xi| i=1,..,N} as input to Prosail and generate {Yi| i=1,2,...,N}

(3) LookUp Table (LUT) Approaches

If the radiative transfer model f(.) is known, we can simulate a collection of X and Y pairs, i.e., $\{(X_j,Y_j) \mid j=1,2,...,M\}$, based on which we can build a LUT and use it to estimate the X value of an observed Y value.

Forward model: $Y = f(X)$, where f(.) is the radiative transfer models, which tend to be highly nonlinear and not invertible.

Based on $Y=f(X)$, we build a LUT by first sampling X_j uniformly within a range [A,B] (A and B are respectively the theoretical min and max value of X), and then obtaining the associated Y_j value by $Y_j=f(X_j)$.

X_1 : Y_1
X_2 : Y_2
X_3 : Y_3
....
X_n : Y_n

Given an observed Y value, how do we estimate its X value using LUT? First, we search the LUT to identify the row whose Y_j value is the closest to the observed Y value. Then, the X_j value associated with Y_j is treated as the estimated X value of Y.

Comparing with the numerical approaches, the LUT approach is simpler and has theoretical advantages such as being able to find the global optimum in the parameter space, and thereby the LUT approaches have been widely used in solving remote sensing inverse problems.

	(1) Direct inversion	**(2) LUT approach**	(3) Numerical Approach	(4) Simulation & ML	(5) ML	(6) DL
f(.) is known	yes	**yes**	yes	yes	yes	yes
f(.) is partially known, i.e., form known, but with some unknown parameters U	no	**no**	Yes, estimate X and U together	no	no	no
f(.) unknown, (X,Y) known	no	**no**	no	no	yes	yes
f(.) unknown, (X,Y) unknown	no	**no**	no	no	no	no
If both f(.) and (X,Y) known, can accommodate both?	no	**yes?**	Yes? Use (X,Y) to estimate parameters in f(.)	yes?	Yes, use both simulated and observed data	Yes, use both simulated and observed data
Can use prior information? e.g., spatial prior and value prior	no	**Yes? Use value prior for sampling**	Yes? Use value prior of X in Bayesian estimation	Yes, Use value prior in sampling and spatial prior in Random fields	Yes, spatial prior in Random field approaches	Yes, similar to ML
Advantages	Knowledge -driven; Simple, easy	**Knowledge-driven; Intuitive, easy, discrete fitting;**	Knowledge-driven; estimate U; Efficient for simple f(.) in convex problems	Knowledge-driven; flexible; continuous fitting; good inter/extrapolation; faster than LUT	Data-driven; flexible; Classic;	Strong modeling capability; automatic feature learning;
Disadvantages	Unrealistic; rely on simple f(.)	**Sensitive to accuracy of f(.), similarity metrics, sampling density and range; slow if LUT is large; bad for extrapolation;**	Rely on efficiency of nonlinear solver; Slow; Local optimum;	Overfitting and underfitting risk to simulated data; difficult model selection; Sensitive to accuracy of f(.), similarity metrics, sampling density and range;	Weak modeling capability; Rely on “good” engineered features; Black-box; Overfitting, underfitting; Feature and model selection is difficult and slow	Overfitting and underfitting; Black-box;

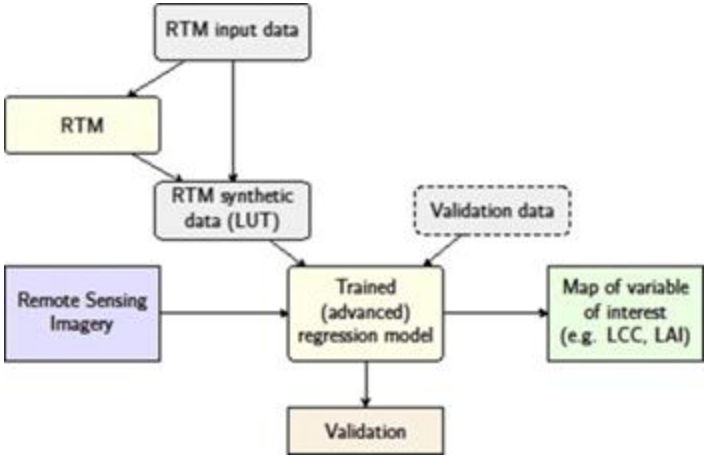
(4) Data Simulation & Machine Learning (ML) Approaches

If the radiative transfer model f(.) is known, we can simulate a collection of X and Y pairs, i.e., $\{(X_j,Y_j) \mid j=1,2,...,M\}$. Instead of using LUT for data inversion, we can use ML approaches to learn the inverse function, i.e., X=g(Y), and use this inverse function to estimate the X value of an observed Y value.

Forward model: Y = f(X), where f(.) is the radiative transfer models, which tend to be highly nonlinear and not invertible.

Based on Y=f(X), similar to LUT, we simulate the following X and Y pairs.

X_1 : Y_1
X_2 : Y_2
X_3 : Y_3
....
X_4 : Y_4



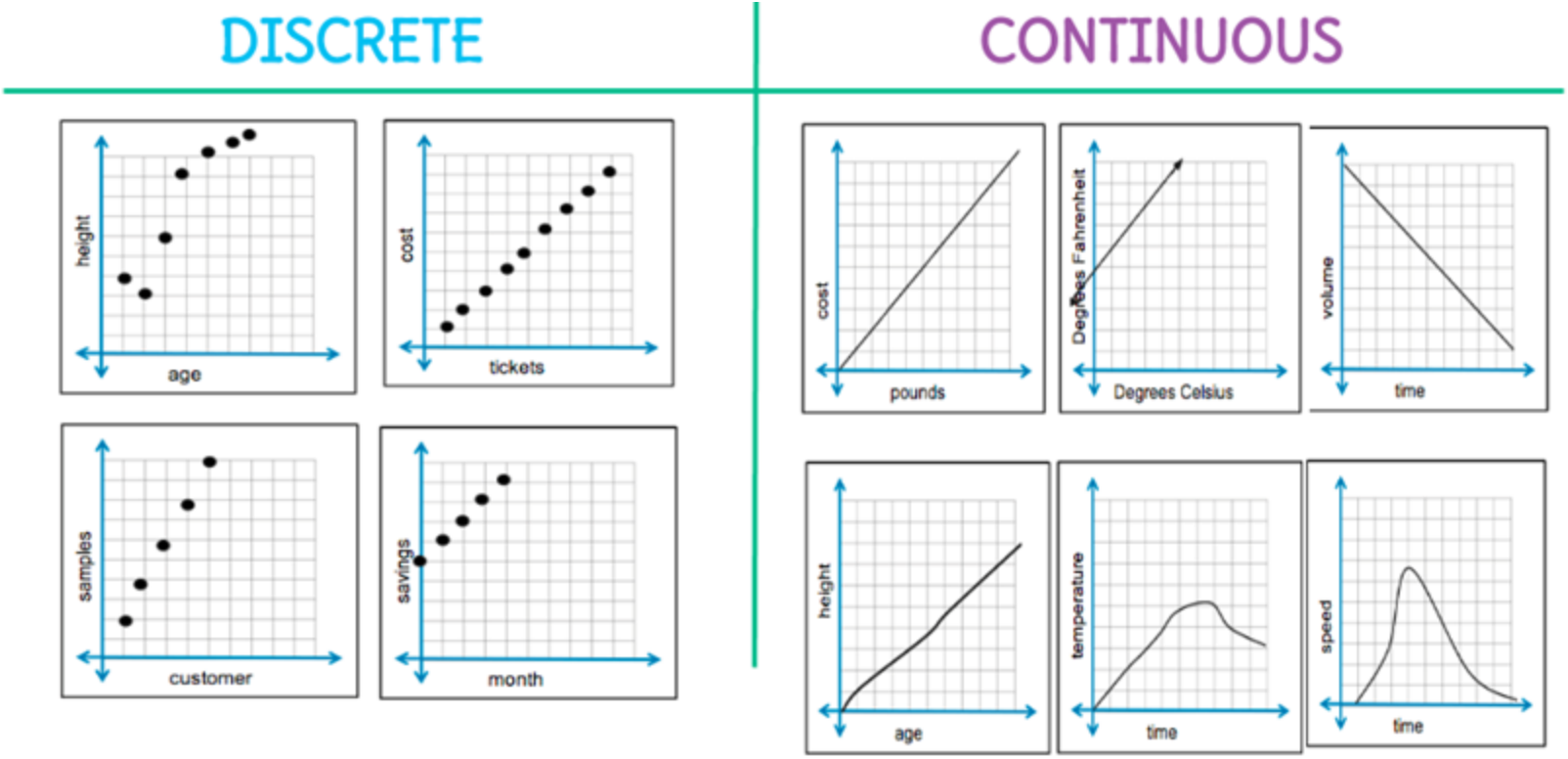
Based on simulated **$\{(X_j,Y_j) \mid j=1,2,...,M\}$** pairs, we build the following objective function:

$J(\theta) = \sum||X_i-g(Y_i)||$
$\theta = \min J(\theta)$

where θ is the unknown parameters in the inverse function g(.) which is a statistical model or machine learning model. Once we know θ, we can establish the inverse function g(.), and use it to estimate the X value of an observed Y value by X=g(Y).

Comparing with the LUT approach that is essentially discrete interpolation, the ML approaches can learn a continuous inverse function g(.) using the simulated data, and thereby they theoretically can achieve more accurate estimation. Moreover, ML approaches tend to be faster because they do not need to do LUT searching for every observation.

LUT vs. Machine leaning



True inverse function vs. approximated inverse function

Forward model:

$Y = f(X)$

(1) Y: received radiation by the sensor

(2) X: variables that you want to know, e.g., class labels, chlorophyll content in leaves, leaf area index/density;

True inverse function:

$X = t(Y) = f^{-1}(Y)$

where $f^{-1}(.)$ is difficult to get and

the form of t(.) is usually unknown;

Approximated inverse function:

$X = g(Y)$

Note that g(.) is only an approximation to the true inverse function t(.)

Appropriate fitting: when the complexity of g(.) is close to t(.);

Overfitting: when the complexity of g(.) is larger than t(.);

Underfitting: when the complexity of g(.) is smaller than t(.);

Based on $\{(X_j,Y_j) \mid j=1,2,...,n\}$, we build the following objective function:

$J(\theta) = \sum||X_i - g(Y_i)||$

$\theta = \min J(\theta)$

Planck's Law

$$B(\lambda,T) = c_1 / \lambda^5 / [e^{c_2/\lambda T} - 1] \quad (W/m^2/ster/um)$$

where λ = wavelengths in um

T = temperature of emitting surface (deg K)

$c_1 = 1.191044 \times 10^{-5}$ $(mW/m^2/ster/cm^{-4})$

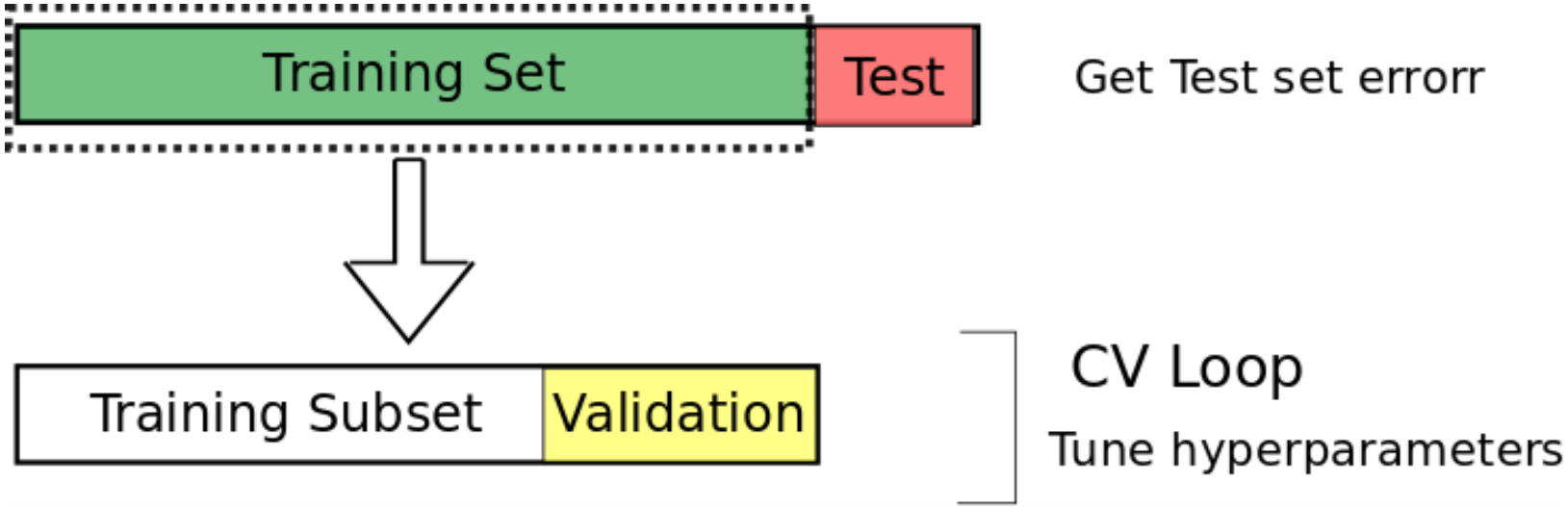
$c_2 = 1.438769$ (cm deg K)

What happens if you fit the BT function using a million-layers neural network?

Brightness Temperature

$$T = c_2 / [\lambda \ln(\frac{c_1}{\lambda^5 B_\lambda} + 1)]$$ is determined by inverting Planck function.

How do you select the “best” g(.)?



Try different models, $g_1(.)$, $g_2(.)$, …, $g_n(2)$, and select the one that with highest accuracy on the validation set.

Overfitting vs. Underfitting

Overfitting:

---- ML model is so flexible and complex that it accommodates the noise effect in the training data and treats it as signal, and the learnt noise characteristics cannot generalize well to the test data;

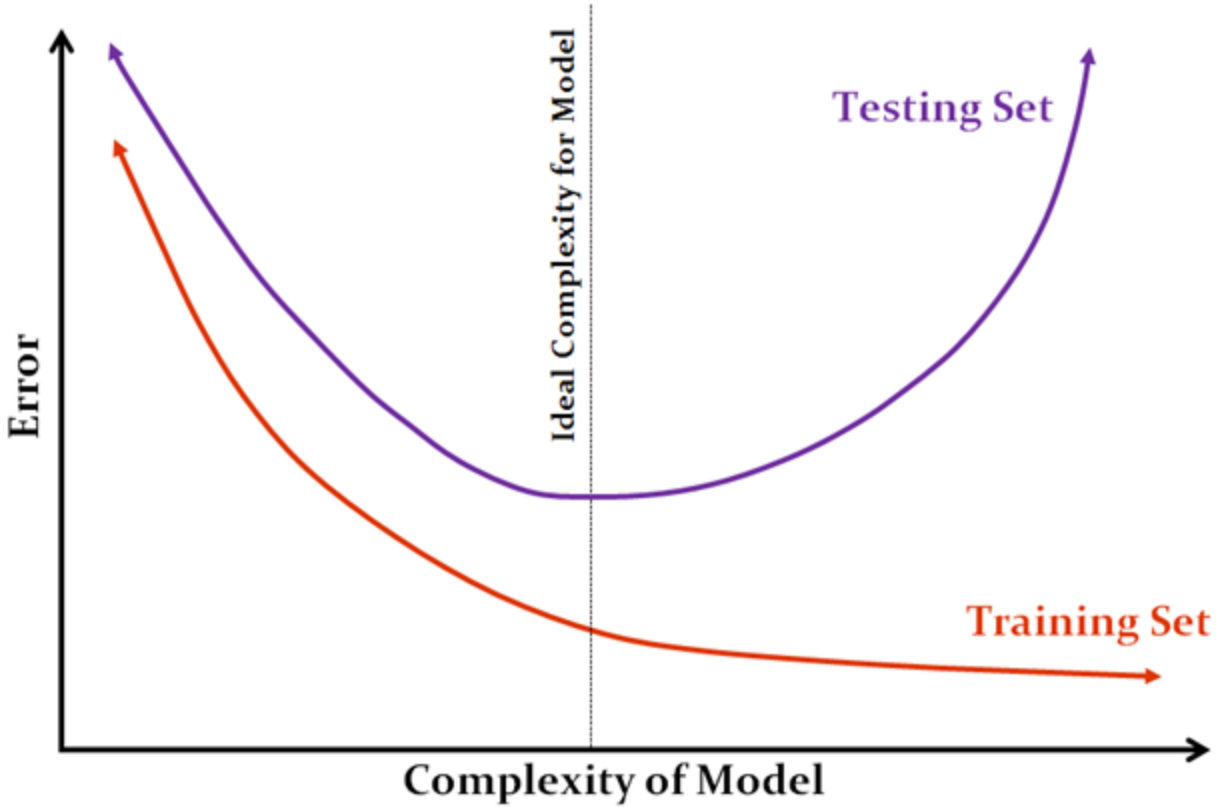
---- very high training accuracy but low validation/test accuracy;

Underfitting:

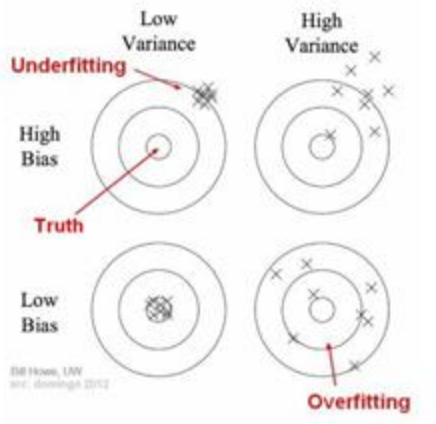
---- ML model is so simple and rigid that it does not has enough capacity to accommodate signal in the training data, and the learnt biased/partial information cannot generalize well to the test data;

---- low training accuracy & low validation/test accuracy;

Training error vs. test error as model complexity changes



What are Bias and Variance? Why overfitting means small bias and big variance?

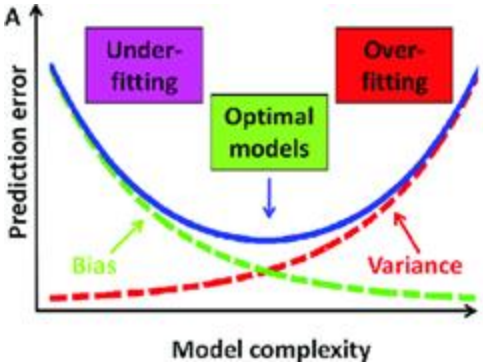


Bias is the difference between the average prediction of our model and the true value which we are trying to predict.

Variance is the variability of model prediction.

Why increasing model complexity lead to small bias in prediction?

---- increasing model complexity -> g(.) to be universal approximator -> stronger accommodating/modeling capability to learn the genuine nonlinear relationship between X and Y in X = t(Y) -> less bias;



Why increasing model complexity lead to larger variance in prediction?

---- increasing model complexity -> g(.) to be universal approximator -> stronger accommodating/modeling capability to learn both the genuine nonlinear relationship between X and Y and irrelevant factors (i.e., noise and even errors in the data) -> larger variance;

Why decreasing model complexity lead to larger bias in prediction?

Why increasing model complexity lead to smaller variance in prediction?

Overfitting vs. Underfitting

Overfitting:

---- ML model is so flexible and complex that it accommodates the noise effect in the training data and treats it as signal, and the learnt noise characteristics cannot generalize well to the test data;

---- very high training accuracy but low validation/test accuracy; small Bias but big variation in prediction;

Underfitting:

---- ML model is so simple and rigid that it does not has enough capacity to accommodate signal in the training data, and the learnt biased/parcial information cannot generalize well to the test data;

---- low training accuracy & low validation/test accuracy; big Bias but small variation in prediction;

	(1) Direct inversion	(2) LUT approach	(3) Numerical Approach	**(4) Simulation & ML**	(5) ML	(6) DL
f(.) is known	yes	yes	yes	**yes**	yes	yes
f(.) is partially known, i.e., form known, but with some unknown parameters U	no	no	Yes, estimate X and U together	**no**	no	no
f(.) unknown, (X,Y) known	no	no	no	**no**	yes	yes
f(.) unknown, (X,Y) unknown	no	no	no	**no**	no	no
If both f(.) and (X,Y) known, can accommodate both?	no	yes?	Yes? Use (X,Y) to estimate parameters in f(.)	**yes?**	Yes, use both simulated and observed data	Yes, use both simulated and observed data
Can use prior information? e.g., spatial prior and value prior	no	Yes? Use value prior for sampling	Yes? Use value prior of X in Bayesian estimation	**Yes, Use value prior in sampling and spatial prior in Random fields**	Yes, spatial prior in Random field approaches	Yes, similar to ML
Advantages	Knowledge -driven; Simple, easy	Knowledge-driven; Intuitive, easy, discrete fitting;	Knowledge-driven; estimate U; Efficient for simple f(.) in convex problems	**Knowledge-driven; flexible; continuous fitting; good inter/extrapolation; faster than LUT**	Data-driven; flexible; Classic;	Strong modeling capability; automatic feature learning;
Disadvantages	Unrealistic; rely on simple f(.)	Sensitive to accuracy of f(.), similarity metrics, sampling density and range; slow if LUT is large; bad for extrapolation;	Rely on efficiency of nonlinear solver; Slow; Local optimum;	**Overfitting and underfitting risk to simulated data; difficult model selection; Sensitive to accuracy of f(.), similarity metrics, sampling density and range;**	Weak modeling capability; Rely on "good" engineered features; Black-box; Overfitting, underfitting; Feature and model selection is difficult and slow	Overfitting and underfitting; Black-box;

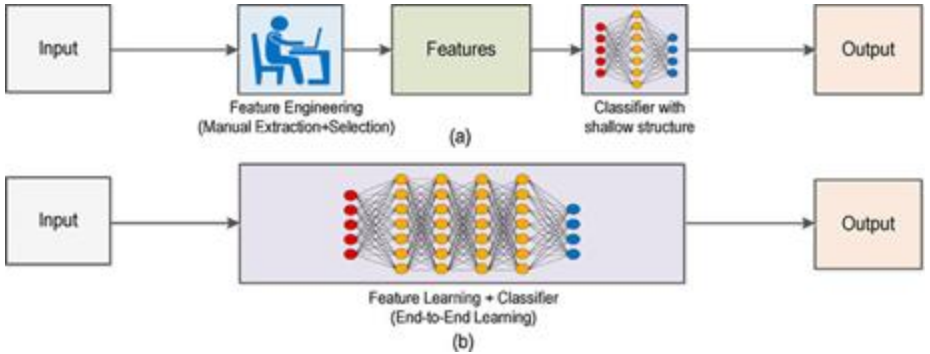
(5) Machine Learning (ML) Approaches

If f(.) is unknown, we resort to data for solving inverse problems. We need to collect X and Y measurements, i.e., $\{(X_j,Y_j) \mid j=1,2,...,T\}$, based on which we establish the inverse function $X=g(Y,\theta)$, where g(.) is a statistical or ML model (empirical models).

No f(X)

We need to measure Y
and the associated ground truth data X to get some (X,Y) pairs,
to train ML model.

X_1 : Y_1
X_2 : Y_2
X_3 : Y_3
....
X_4 : Y_4



Based on **$\{(X_j,Y_j) \mid j=1,2,...,T\}$**, we build the following objective function:

$J(\theta) = \sum \|X_i - g(Y_i)\|$
$\theta = \min J(\theta)$

where θ is the unknown parameters in g(.). Once we know θ, we can establish the inverse function g(.), and use it to estimate the X value of an observed Y value by X=g(Y).

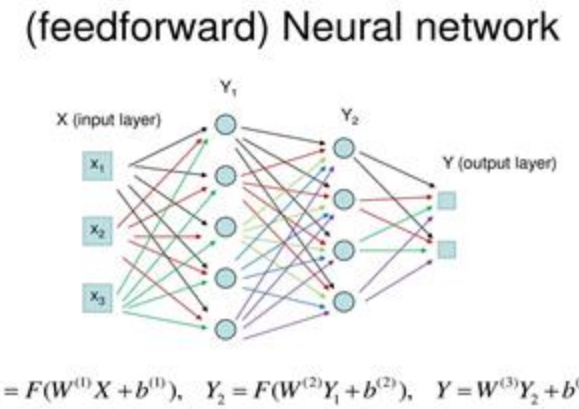
Limitations of ML approaches, when X and Y have highly-nonlinear relationship? Why feature extraction and selection? Why not using original features?

	(1) Direct inversion	(2) LUT approach	(3) Numerical Approach	(4) Simulation & ML	**(5) ML**	(6) DL
f(.) is known	yes	yes	yes	yes	**yes**	yes
f(.) is partially known, i.e., form known, but with some unknown parameters U	no	no	Yes, estimate X and U together	no	**no**	no
f(.) unknown, (X,Y) known	no	no	no	no	**yes**	yes
f(.) unknown, (X,Y) unknown	no	no	no	no	**no**	no
If both f(.) and (X,Y) known, can accommodate both?	no	yes?	Yes? Use (X,Y) to estimate parameters in f(.)	yes?	**Yes, use both simulated and observed data**	Yes, use both simulated and observed data
Can use prior information? e.g., spatial prior and value prior	no	Yes? Use value prior for sampling	Yes? Use value prior of X in Bayesian estimation	Yes, Use value prior in sampling and spatial prior in Random fields	**Yes, spatial prior in Random field approaches**	Yes, similar to ML
Advantages	Knowledge -driven; Simple, easy	Knowledge-driven; Intuitive, easy, discrete fitting;	Knowledge-driven; estimate U; Efficient for simple f(.) in convex problems	Knowledge-driven; flexible; continuous fitting; good inter/extrapolation; faster than LUT	**Data-driven; flexible; Classic;**	Strong modeling capability; automatic feature learning;
Disadvantages	Unrealistic; rely on simple f(.)	Sensitive to accuracy of f(.), similarity metrics, sampling density and range; slow if LUT is large; bad for extrapolation;	Rely on efficiency of nonlinear solver; Slow; Local optimum;	Overfitting and underfitting risk to simulated data; difficult model selection; Sensitive to accuracy of f(.), similarity metrics, sampling density and range;	**Weak modeling capability; Rely on "good" engineered features; Black-box; Overfitting, underfitting; Feature and model selection is difficult and slow**	Overfitting and underfitting; Black-box;

(6) Deep Learning (DL) Approaches

Deep learning (DL) approaches are also ML approaches, and as such they can be used for data inversion through (4) and (5), i.e.,

--- if f(.) is known, we simulate $\{(X_j,Y_j) \mid j=1,2,...,T\}$ using f(.) and use them to train DL models for obtaining the inverse function X=g(Y);
--- if f(.) is unknown, we obtain remote sensing data Y and ground truth data X to build X and Y pairs, i.e., $\{(X_j,Y_j) \mid j=1,2,...,T\}$, and use them to train DL models for obtaining the inverse function X=g(Y);



Based on **training data**, we build the following objective function:

$J(\theta) = \sum \|X_i - g(Y_i)\|$
$\theta = \min J(\theta)$

where θ is the unknown parameters in DL model g(.). Once we know θ, we can establish the inverse function g(.), and use it to estimate the X value of an observed Y value by X=g(Y).

Comparing with traditional ML approaches, such as SVM and random forest, the DL approaches, due to their strong modeling capability and GPU computation, are more capable of effectively and efficiently learning the complex nonlinear relationship between Y and X, and perform accurate and fast model prediction for estimating X.

When DL is better than ML?

Forward model:

$Y = f(X)$

(1) Y: received radiation by the sensor

(2) X: variables that you want to know, e.g., class labels, chlorophyll content in leaves, leaf area index/density;

True inverse function:

$X = t(Y) = f^{-1}(Y)$

where $f^{-1}(.)$ is difficult to get and

the form of t(.) is usually unknown;

Approximated inverse function:

$X = g(Y)$

Note that g(.) is only an approximation to the true inverse function t(.)

Appropriate fitting: when the complexity of g(.) is close to t(.);

Overfitting: when the complexity of g(.) is larger than t(.);

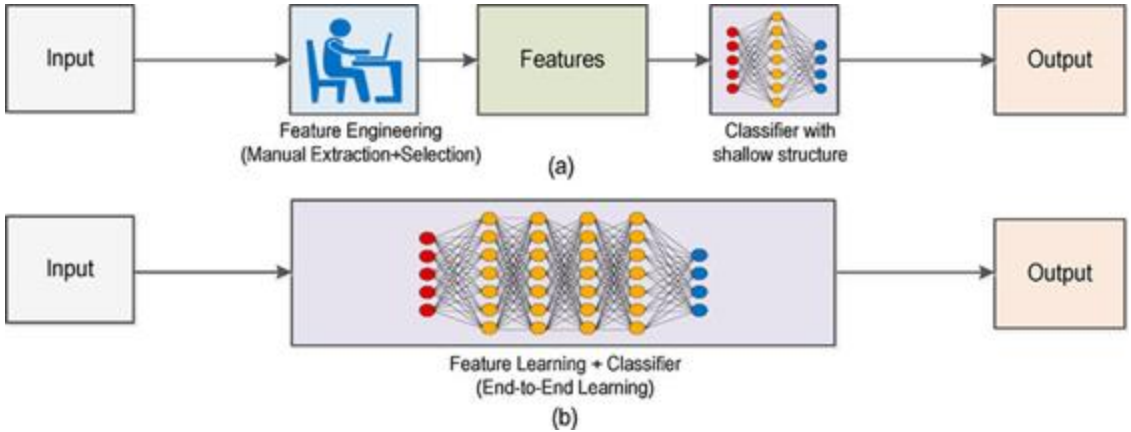
Underfitting: when the complexity of g(.) is smaller than t(.);

Based on $\{(X_j,Y_j) \mid j=1,2,...,n\}$, we build the following objective function:

$J(\theta) = \sum||X_i - g(Y_i)||$

$\theta = \min J(\theta)$

Feature-driven machine learning vs. Data-driven deep learning (DL)



Advantages of DL approaches for RS image classification:

(1) automatically learn the “best” feature without requiring task-specific classifier-specific knowledge;

(2) End-to-end approach without any intermediate stages in the data-processing pipeline;

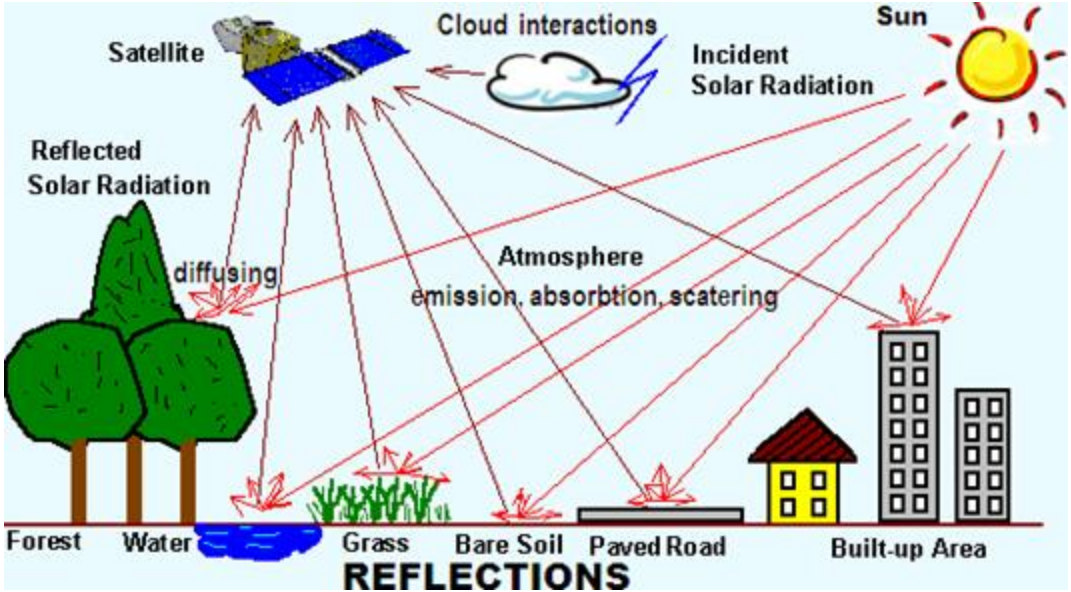
(3) Complex model -> strong modeling capability -> efficiently capture the subtle differences among classes;

(4) Powerful GPU computation

	(1) Direct inversion	(2) LUT approach	(3) Numerical Approach	(4) Simulation & ML	(5) ML	**(6) DL**
f(.) is known	yes	yes	yes	yes	yes	**yes**
f(.) is partially known, i.e., form known, but with some unknown parameters U	no	no	Yes, estimate X and U together	no	no	**no**
f(.) unknown, (X,Y) known	no	no	no	no	yes	**yes**
f(.) unknown, (X,Y) unknown	no	no	no	no	no	**no**
If both f(.) and (X,Y) known, can accommodate both?	no	yes?	Yes? Use (X,Y) to estimate parameters in f(.)	yes?	Yes, use both simulated and observed data	**Yes, use both simulated and observed data**
Can use prior information? e.g., spatial prior and value prior	no	Yes? Use value prior for sampling	Yes? Use value prior of X in Bayesian estimation	Yes, Use value prior in sampling and spatial prior in Random fields	Yes, spatial prior in Random field approaches	**Yes, similar to ML**
Advantages	Knowledge -driven; Simple, easy	Knowledge-driven; Intuitive, easy, discrete fitting;	Knowledge-driven; estimate U; Efficient for simple f(.) in convex problems	Knowledge-driven; flexible; continuous fitting; good inter/extrapolation; faster than LUT	Data-driven; flexible; Classic;	**Strong modeling capability; automatic feature learning;**
Disadvantages	Unrealistic; rely on simple f(.)	Sensitive to accuracy of f(.), similarity metrics, sampling density and range; slow if LUT is large; bad for extrapolation;	Rely on efficiency of nonlinear solver; Slow; Local optimum;	Overfitting and underfitting risk to simulated data; difficult model selection; Sensitive to accuracy of f(.), similarity metrics, sampling density and range;	Weak modeling capability; Rely on "good" engineered features; Black-box; Overfitting, underfitting; Feature and model selection is difficult and slow	**Overfitting and underfitting; Black-box;**

More on Error and Prior

Remote Sensing System Overview



Forward model:

$Y = f(X)$

(1) Y: radiation received by the sensor

(2) X: variables that you want to estimate, e.g., class labels, chlorophyll content in leaves, leaf area index/density;

Inverse model:

$X = g(Y, \theta)$

where g(.) is an unknown inverse function with unknown model parameter θ.

Why estimating X is difficult?

--- Knowledge f(.) complex, biased, highly nonlinear, with large uncertainty;

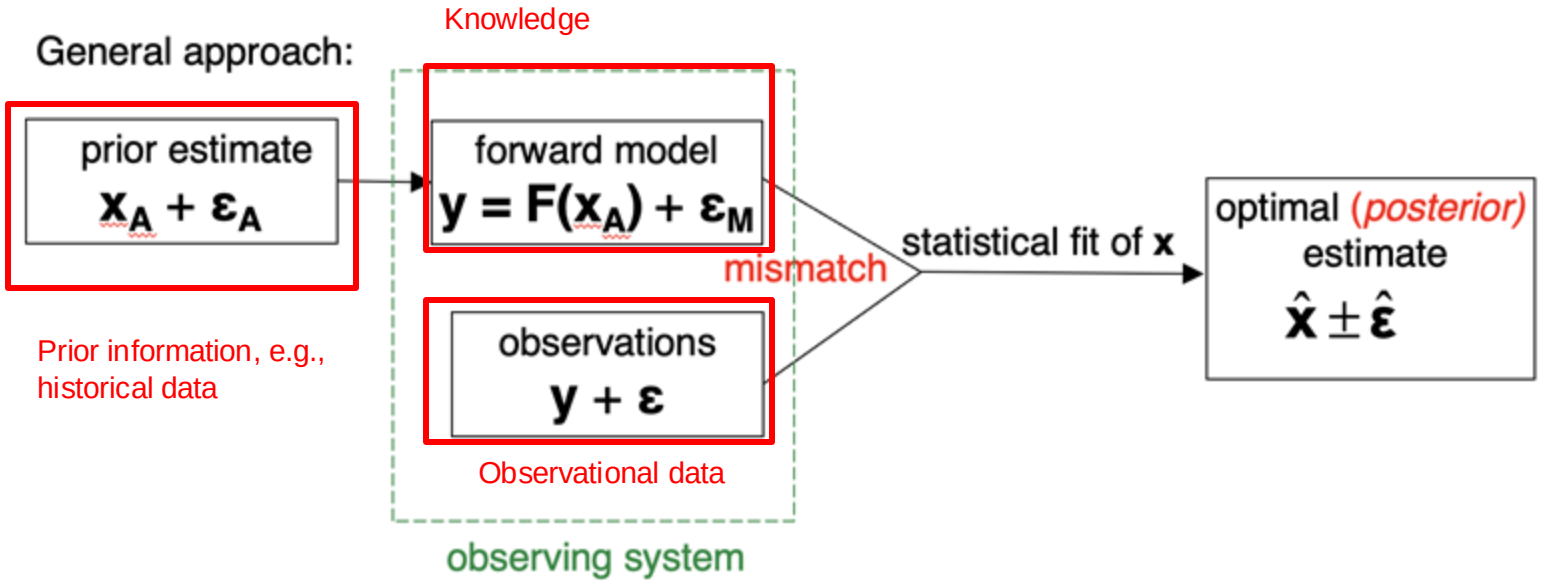
---- Data (X, Y) pairs limited, poor quality;

---- Prior information (e.g., spatial prior) ambiguous;

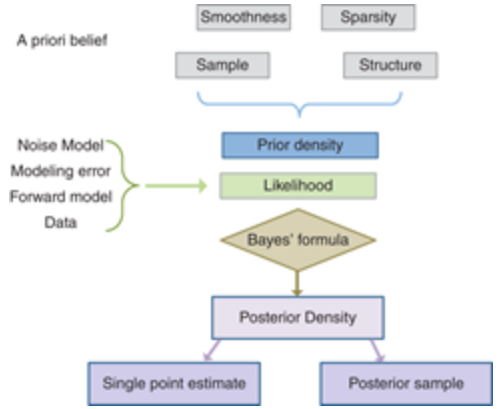
How to use prior estimate (e.g., historical estimate) and accommodate errors?

Estimate selected variables driving a physical system (*state vector* **x**, dim *n*) by using:

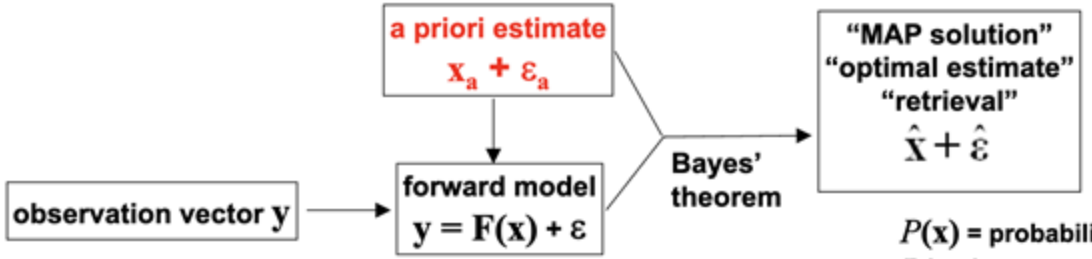
- the observable manifestations of the system (*observation vector* **y**, dim *m*)
- a physical model $\mathbf{y} = \mathbf{F}(\mathbf{x})$ (*forward model*)
- a *prior estimate* $\mathbf{x}_A$ before the observations have been made
- Statistics for the errors $\boldsymbol{\varepsilon}$ in the different components of the problem



Bayesian framework - use "prior" & conditional distributions



Optimize values of an ensemble of variables (*state vector* x) using observations:

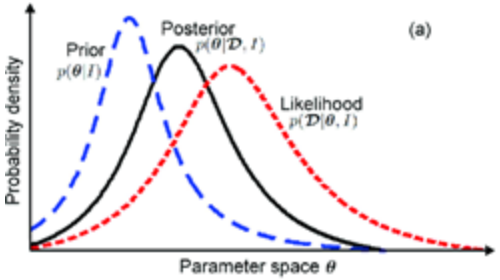


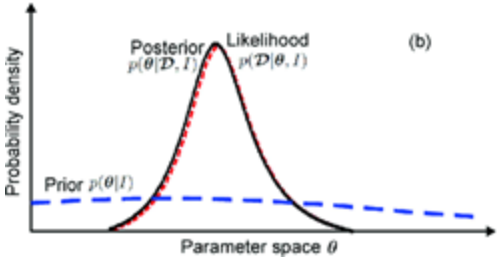
$P(\mathbf{x})$ = probability distribution function (pdf) of x

$P(\mathbf{x},\mathbf{y})$ = pdf of (x,y)

$P(\mathbf{y}|\mathbf{x})$ = pdf of y given x

$$P(\mathbf{x},\mathbf{y})d\mathbf{x}d\mathbf{y} \begin{cases} = P(\mathbf{x})d\mathbf{x}P(\mathbf{y}\,|\,\mathbf{x})d\mathbf{y} \\ = P(\mathbf{y})d\mathbf{y}P(\mathbf{x}\,|\,\mathbf{y})d\mathbf{x} \end{cases}$$





a posteriori pdf | observation *pdf* | *a priori* pdf

$$\Rightarrow \quad P(\mathbf{x}\,|\,\mathbf{y}) = \frac{P(\mathbf{y}\,|\,\mathbf{x})P(\mathbf{x})}{P(\mathbf{y})}$$

Bayes' theorem

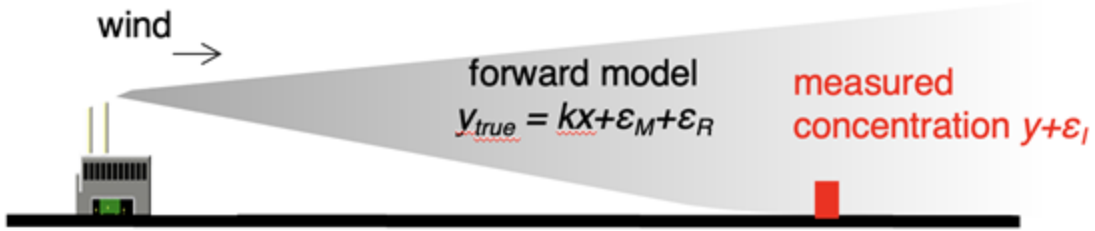
normalizing factor (unimportant)

Maximum *a posteriori* (MAP) solution for x given y is defined by $\max(P(\mathbf{x}\,|\,\mathbf{y}))$

⇨ **solve for** $\nabla_{\mathbf{x}} P(\mathbf{x}\,|\,\mathbf{y}) = \mathbf{0}$

A simple linear inverse problem

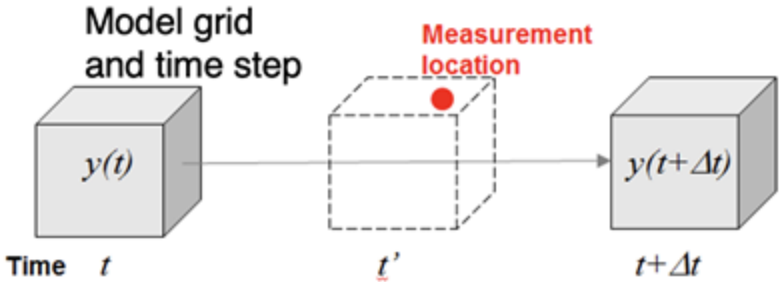
Optimize estimate of true emission x using a single concentration measurement y; assume simple linear transport forward model described by $y = kx$



prior emission $x_A+\varepsilon_A$ error variance $\sigma_A^2 = E[\varepsilon_A^2]$ expected value operator

Instrument error ε_I
Forward model error ε_M

Representation error ε_R:



Observational error $\varepsilon_O = \varepsilon_I + \varepsilon_M + \varepsilon_R$ $y = kx + \varepsilon_O$ error variance $\sigma_O^2 = E[\varepsilon_O^2]$

Error variances are additive: $\sigma_O^2 = \sigma_I^2 + \sigma_M^2 + \sigma_R^2$

How to solve this inverse problem using Bayesian MAP estimation? error distribution assumption?

Prior pdf: $$P(x) = \frac{1}{\sigma_A \sqrt{2\pi}} \exp[-\frac{(x - x_A)^2}{2\sigma_A^2}]$$

Observation pdf: $$P(y \mid x) = \frac{1}{\sigma_O \sqrt{2\pi}} \exp[-\frac{(y - kx)^2}{2\sigma_O^2}]$$

Posterior pdf: $$P(x \mid y) \propto P(x)P(y \mid x) \propto \exp[-\frac{(x - x_A)^2}{2\sigma_A^2} - \frac{\quad}{2\sigma_O^2}]$$

Cost function to minimize: $$J(x) = \frac{(x - x_A)^2}{\sigma_A^2} + \frac{(y - kx)^2}{\sigma_O^2}$$ solve $dJ/dx = 0$

Posterior solution: $$\hat{x} = x_A + g(y - kx_A) \qquad g = \frac{k\sigma_A^2}{k^2\sigma_A^2 + \sigma_O^2}$$ gain factor

with posterior pdf $$P(x \mid y) = \frac{1}{\hat{\sigma}\sqrt{2\pi}} \exp[-\frac{(x - \hat{x})^2}{2\hat{\sigma}^2}] \qquad \frac{1}{\hat{\sigma}^2} = \frac{1}{\sigma_A^2} + \frac{1}{(\sigma_O / k)^2}$$

posterior error variance

Relationship to truth: $$\hat{x} = x + (1 - a)(x_A - x) + g\varepsilon_O \qquad a = gk = \frac{\sigma_A^2}{\sigma_A^2 + (\sigma_O / k)^2}$$

optimal estimate | truth | smoothing error | obs. error

averaging kernel

How to solve this inverse problem using Bayesian MAP estimation?

Prior pdf: $$P(x) = \frac{1}{\sigma_A \sqrt{2\pi}} \exp[-\frac{(x - x_A)^2}{2\sigma_A^2}]$$

Observation pdf: $$P(y \mid x) = \frac{1}{\sigma_O \sqrt{2\pi}} \exp[-\frac{(y - kx)^2}{2\sigma_O^2}]$$

Posterior pdf: $$P(x \mid y) \propto P(x)P(y \mid x) \propto \exp[-\frac{(x - x_A)^2}{2\sigma_A^2} - \frac{(y - kx)^2}{2\sigma_O^2}]$$

Cost function to minimize: $$J(x) = \frac{(x - x_A)^2}{\sigma_A^2} + \frac{(y - kx)^2}{\sigma_O^2}$$ solve $dJ/dx = 0$

Posterior solution: $$\hat{x} = x_A + g(y - kx_A) \qquad g = \frac{k\sigma_A^2}{k^2\sigma_A^2 + \sigma_O^2}$$ gain factor

with posterior pdf $$P(x \mid y) = \frac{1}{\hat{\sigma}\sqrt{2\pi}} \exp[-\frac{(x - \hat{x})^2}{2\hat{\sigma}^2}] \qquad \frac{1}{\hat{\sigma}^2} = \frac{1}{\sigma_A^2} + \frac{1}{(\sigma_O / k)^2}$$

posterior error variance

Relationship to truth: $$\hat{x} = x + (1 - a)(x_A - x) + g\varepsilon_O \qquad a = gk = \frac{\sigma_A^2}{\sigma_A^2 + (\sigma_O / k)^2}$$

optimal estimate | truth | smoothing error | obs. error

averaging kernel

How to solve this inverse problem using Bayesian MAP estimation?

Instead of a single measurement *y*, make *m* measurements $\mathbf{y} = (y_1, \ldots y_m)^T$

Cost function:

mean

$$J(x) = \frac{(x - x_A)^2}{\sigma_A^2} + \sum_{i=1}^{m} \frac{(y_i - kx)^2}{\sigma_O^2} = \frac{(x - x_A)^2}{\sigma_A^2} + \frac{\overline{(y_i - kx)^2}}{\sigma_O^2 / m}$$

vs.

$$J(x) = \frac{(x - x_A)^2}{\sigma_A^2} + \frac{(y - kx)^2}{\sigma_O^2}$$

Observational error variance decreases as *m* (central limit theorem) IF:

- Error is random
- Individual measurements are not correlated and sample the same pdf

If there is systematic bias in observing system, the bias will propagate to the solution:

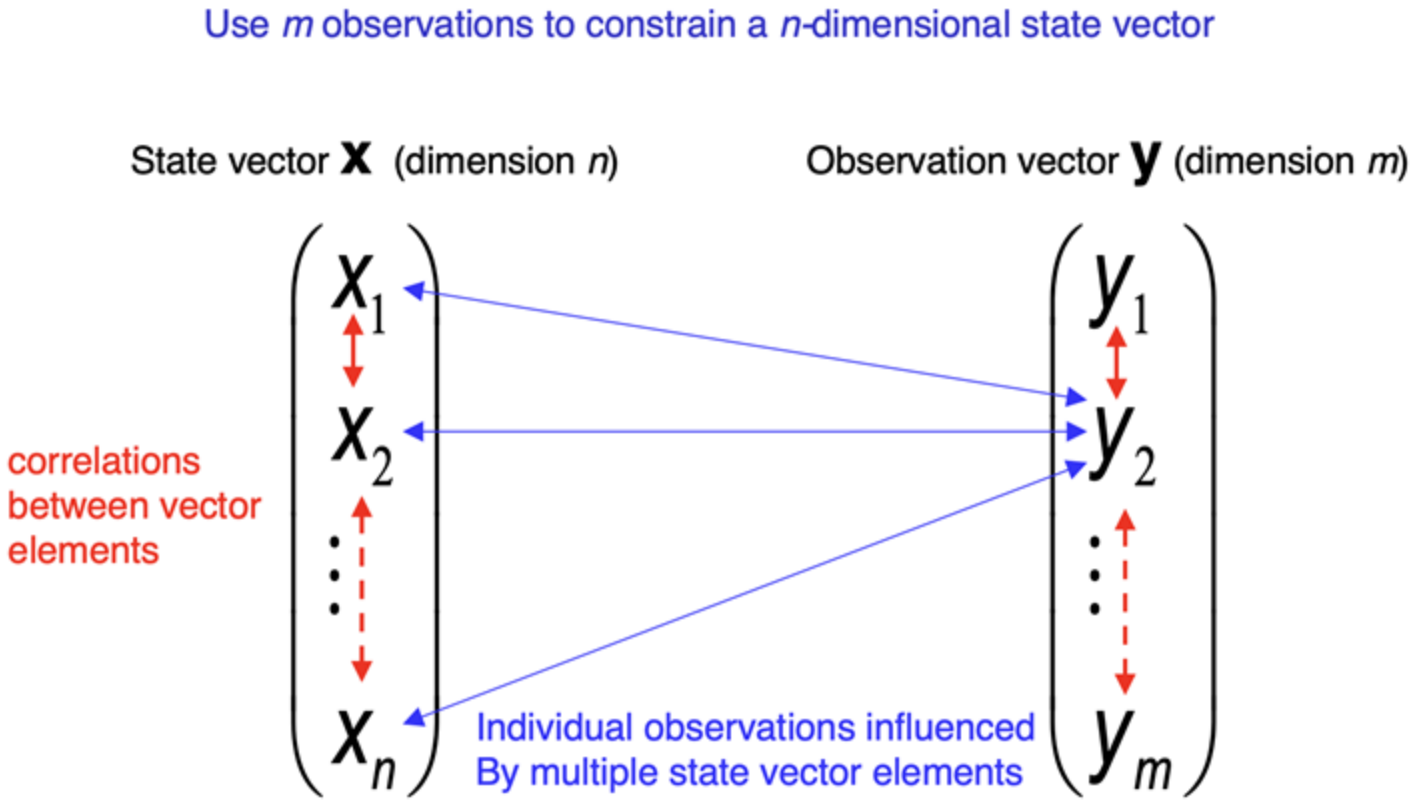
$$y = kx + b_O + \varepsilon'_O \quad \Rightarrow \quad \hat{x} = x + (1 - a)(x_A - x) + g\varepsilon'_O + gb_O$$

bias, residual error

truth, smoothing error, obs. error, bias

Removing bias by validation of the observing system is critical!

How to formulate this inverse problem using vectors?



How to formulate error variance matrices and PDFs for the vectors?

Consider vector **x** with expected value $E[\mathbf{x}]$ and error $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots \varepsilon_n)^T$

Error covariance matrix **S**:

$$\mathbf{S} = \begin{pmatrix} \mathrm{var}(\varepsilon_1) & \cdots & \mathrm{cov}(\varepsilon_1, \varepsilon_n) \\ \vdots & \ddots & \vdots \\ \mathrm{cov}(\varepsilon_1, \varepsilon_n) & \cdots & \mathrm{var}(\varepsilon_n) \end{pmatrix}$$

Gaussian error PDF:

$$P(\mathbf{x}) = \frac{1}{(2\pi)^{n/2} |\mathbf{S}|^{1/2}} \exp[-\frac{1}{2}(\mathbf{x} - \square)^T \mathbf{S}^{-1}(\mathbf{x} - \square)]$$

We will make use of

$$\ln P(\mathbf{x}) \propto -(\mathbf{x} - \square)^T \mathbf{S}^{-1}(\mathbf{x} - \square)]$$

How to formulate error variance matrices and PDFs for the vectors?

Consider vector **x** with expected value $E[\mathbf{x}]$ and error $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots \varepsilon_n)^T$

Error covariance matrix **S**:

$$\mathbf{S} = \begin{pmatrix} \mathrm{var}(\varepsilon_1) & \cdots & \mathrm{cov}(\varepsilon_1, \varepsilon_n) \\ \vdots & \ddots & \vdots \\ \mathrm{cov}(\varepsilon_1, \varepsilon_n) & \cdots & \mathrm{var}(\varepsilon_n) \end{pmatrix}$$

Gaussian error PDF:

$$P(\mathbf{x}) = \frac{1}{(2\pi)^{n/2} |\mathbf{S}|^{1/2}} \exp[-\frac{1}{2}(\mathbf{x} - E[\mathbf{x}])^T \mathbf{S}^{-1} (\mathbf{x} - E[\mathbf{x}])]$$

We will make use of

$$\ln P(\mathbf{x}) \propto -(\mathbf{x} - E[\mathbf{x}])^T \mathbf{S}^{-1} (\mathbf{x} - E[\mathbf{x}])]$$

How to formulate Bayesian MAP estimation using vectors?

State vector $\mathbf{x} = (x_1, \ldots x_n)^T$ Obs vector $\mathbf{y} = (y_1, \ldots y_m)^T$ Forward model $\mathbf{y} = \mathbf{F}(\mathbf{x})$

Prior $\mathbf{x_A}$
with prior error covariance matrix $\mathbf{S_A}$

Observation $\mathbf{y}$
with observational error covariance matrix $\mathbf{S_O}$

prior pdf $P(\mathbf{x})$

$$\ln P(\mathbf{x}) \propto -(\mathbf{x} - \mathbf{x_A})^T \mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A})$$

observational pdf $P(\mathbf{y} \mid \mathbf{x})$

$$\ln P(\mathbf{y} \mid \mathbf{x}) \propto -(\mathbf{y} - \mathbf{F}(\mathbf{x}))^T \mathbf{S_O}^{-1} (\mathbf{y} - \mathbf{F}(\mathbf{x}))$$

$$\ln P(\mathbf{x} \mid \mathbf{y}) \propto \boxed{} + \ln P(\mathbf{y} \mid \mathbf{x}) = \boxed{} - (\mathbf{y} - \mathbf{F}(\mathbf{x}))^T \mathbf{S_O}^{-1} (\mathbf{y} - \mathbf{F}(\mathbf{x}))$$

Minimize cost function: $J(\mathbf{x}) = \boxed{} + (\mathbf{y} - \mathbf{F}(\mathbf{x}))^T \mathbf{S_O^{-1}} (\mathbf{y} - \mathbf{F}(\mathbf{x}))$

$\Longrightarrow$ solve $\nabla_x J(\mathbf{x}) = 2\mathbf{S_A^{-1}}(\mathbf{x} - \mathbf{x_A}) + 2(\nabla_x \mathbf{F})^T \mathbf{S_O^{-1}}(\mathbf{F}(\mathbf{x}) - \mathbf{y}) = \mathbf{0}$

How to formulate Bayesian MAP estimation using vectors?

State vector $\mathbf{x} = (x_1, \ldots x_n)^T$ Obs vector $\mathbf{y} = (y_1, \ldots y_m)^T$ Forward model $\mathbf{y} = \mathbf{F}(\mathbf{x})$

Prior $\mathbf{x_A}$
with prior error covariance matrix $\mathbf{S_A}$

Observation $\mathbf{y}$
with observational error covariance matrix $\mathbf{S_O}$

prior pdf $P(\mathbf{x})$

$$\ln P(\mathbf{x}) \propto -(\mathbf{x} - \mathbf{x_A})^T \mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A})$$

observational pdf $P(\mathbf{y} \mid \mathbf{x})$

$$\ln P(\mathbf{y} \mid \mathbf{x}) \propto -(\mathbf{y} - \mathbf{F}(\mathbf{x}))^T \mathbf{S_O}^{-1} (\mathbf{y} - \mathbf{F}(\mathbf{x}))$$

$$\ln P(\mathbf{x} \mid \mathbf{y}) \propto \ln P(\mathbf{x}) + \ln P(\mathbf{y} \mid \mathbf{x}) = -(\mathbf{x} - \mathbf{x_A})^T \mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A}) - (\mathbf{y} - \mathbf{F}(\mathbf{x}))^T \mathbf{S_O}^{-1} (\mathbf{y} - \mathbf{F}(\mathbf{x}))$$

Minimize cost function: $J(\mathbf{x}) = (\mathbf{x} - \mathbf{x_A})^T \mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A}) + (\mathbf{y} - \mathbf{F}(\mathbf{x}))^T \mathbf{S_O^{-1}} (\mathbf{y} - \mathbf{F}(\mathbf{x}))$

$\Longrightarrow$ solve $\nabla_x J(\mathbf{x}) = 2\mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A}) + 2 (\nabla_x \mathbf{F})^T \mathbf{S_O^{-1}} (\mathbf{F}(\mathbf{x}) - \mathbf{y}) = \mathbf{0}$

Linear forward model leads to an analytical (closed-form) solution

$\mathbf{y} = \mathbf{F(x)} = \mathbf{Kx}$ where $\mathbf{K} = \begin{pmatrix} \partial y_1 / \partial x_1 & \cdots & \partial y_1 / \partial x_n \\ \vdots & \ddots & \vdots \\ \partial y_m / \partial x_1 & \cdots & \partial y_m / \partial x_n \end{pmatrix}$ Is the Jacobian matrix

Then equation from previous slide becomes

$$\nabla_x J(\mathbf{x}) = 2\mathbf{S}_A^{-1}(\mathbf{x} - \mathbf{x}_A) + 2\mathbf{K}^T\mathbf{S}_O^{-1}(\mathbf{Kx} - \mathbf{y}) = \mathbf{0}$$

Solution: $\hat{\mathbf{x}} = \mathbf{x}_A + \mathbf{G}(\mathbf{y} - \mathbf{Kx}_A)$ with $\mathbf{G} = \mathbf{S}_A\mathbf{K}^T(\mathbf{KS}_A\mathbf{K}^T + \mathbf{S}_O)^{-1}$

gain matrix

$$\hat{\mathbf{S}} = (\mathbf{K}^T\mathbf{S}_O^{-1}\mathbf{K} + \mathbf{S}_A^{-1})^{-1}$$

posterior error covariance matrix

Relate solution to true value:

$$\hat{\mathbf{x}} = \mathbf{x} + (\mathbf{I}_n - \mathbf{A})(\mathbf{x}_A - \mathbf{x}) + \mathbf{G}\boldsymbol{\varepsilon}_O \qquad \text{with } \mathbf{A} = \mathbf{GK}$$

truth — smoothing error — observational error — averaging kernel matrix

The importance of the averaging kernel matrix A

A describes the sensitivity of the retrieval to the true state

$$\mathbf{A} = \frac{\partial \hat{\mathbf{x}}}{\partial \mathbf{x}} = \begin{pmatrix} \partial \hat{x}_1 / \partial x_1 & \cdots & \partial \hat{x}_1 / \partial x_n \\ \vdots & \ddots & \vdots \\ \partial \hat{x}_n / \partial x_1 & \cdots & \partial \hat{x}_n / \partial x_n \end{pmatrix}$$

so that $\mathbf{I_n} - \mathbf{A}$ describes the smoothing of the solution:

$$\hat{\mathbf{x}} = \mathbf{x} + (\mathbf{I_n} - \mathbf{A})(\mathbf{x_A} - \mathbf{x}) + \mathbf{G}\boldsymbol{\varepsilon_O}$$

truth | smoothing error | observational error

If A is very small, it means that the observing system is not sufficient for constraining the state vector, meaning that there could be multiple solutions.

Analytical inversion gives **A** as part of the solution:

$$\mathbf{A} = \mathbf{GK} = \mathbf{S_A K}^T (\mathbf{K S_A K}^T + \mathbf{S_O})^{-1} \mathbf{K} = \mathbf{I_n} - \hat{\mathbf{S}} \mathbf{S_A}^{-1}$$

The trace of **A** gives the # of independent pieces of information in the inversion – also called the degrees of freedom for signal (DOFS)

Before making any observations, **A** can diagnose the utility of an observing system for constraining the state vector

How to calculate K using nonlinear forward model?

Constructing the Jacobian matrix K of a forward model y = F(x)

$$\mathbf{K} = \nabla_{\mathbf{x}}\mathbf{F} = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{pmatrix} \partial y_1 / \partial x_1 & \cdots & \partial y_1 / \partial x_n \\ \vdots & \ddots & \vdots \\ \partial y_m / \partial x_1 & \cdots & \partial y_m / \partial x_n \end{pmatrix}$$

two passes of forward model

Jacobian matrix expresses the sensitivity of elements in ***y*** to elements in ***x***.

$$\frac{\partial \mathbf{y}}{\partial x_1} \approx \frac{\mathbf{F}(\mathbf{x_A} + (\Delta x_1, 0, \ldots 0)^T) - \mathbf{F}(\mathbf{x_A})}{\Delta x_1}$$

Constructing Jacobian matrix requires $n + 1$ passes of forward model

How to integrate neural network into this framework? Replace x_A?

State vector $\mathbf{x} = (x_1, \ldots x_n)^T$ Obs vector $\mathbf{y} = (y_1, \ldots y_m)^T$ Forward model $\mathbf{y} = \mathbf{F(x)}$

Prior $\mathbf{x_A}$
with prior error covariance matrix $\mathbf{S_A}$

Observation $\mathbf{y}$
with observational error covariance matrix $\mathbf{S_O}$

prior pdf $P(\mathbf{x})$

$$\ln P(\mathbf{x}) \propto -(\mathbf{x} - \mathbf{x_A})^T \mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A})$$

observational pdf $P(\mathbf{y} \mid \mathbf{x})$

$$\ln P(\mathbf{y} \mid \mathbf{x}) \propto -(\mathbf{y} - \mathbf{F(x)})^T \mathbf{S_O}^{-1} (\mathbf{y} - \mathbf{F(x)})$$

$$\ln P(\mathbf{x} \mid \mathbf{y}) \propto \ln P(\mathbf{x}) + \ln P(\mathbf{y} \mid \mathbf{x}) = -(\mathbf{x} - \mathbf{x_A})^T \mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A}) - (\mathbf{y} - \mathbf{F(x)})^T \mathbf{S_O}^{-1} (\mathbf{y} - \mathbf{F(x)})$$

Minimize cost function: $J(\mathbf{x}) = (\mathbf{x} - \mathbf{x_A})^T \mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A}) + (\mathbf{y} - \mathbf{F(x)})^T \mathbf{S_O^{-1}} (\mathbf{y} - \mathbf{F(x)})$

$\Longrightarrow$ solve $\nabla_x J(\mathbf{x}) = 2\mathbf{S_A^{-1}} (\mathbf{x} - \mathbf{x_A}) + 2 (\nabla_x \mathbf{F})^T \mathbf{S_O^{-1}} (\mathbf{F(x)} - \mathbf{y}) = \mathbf{0}$

Questions?