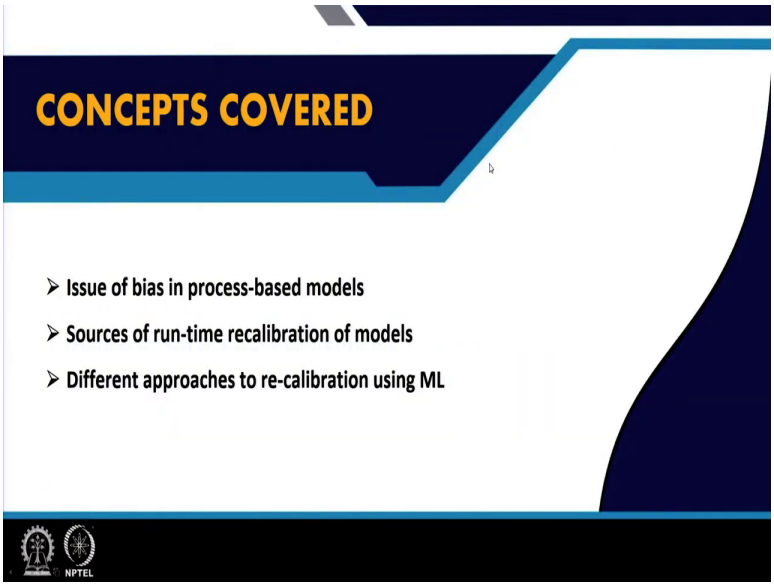
Machine Learning for Earth System Sciences
Prof. Adway Mitra
Department of Computer Science and Engineering
Centre of Excellence in Artificial Intelligence
Indian Institute of Technology, Kharagpur

Module - 05
Machine Learning for Earth System Modelling
Lecture - 39
Data Assimilation for Earth System Model Correction

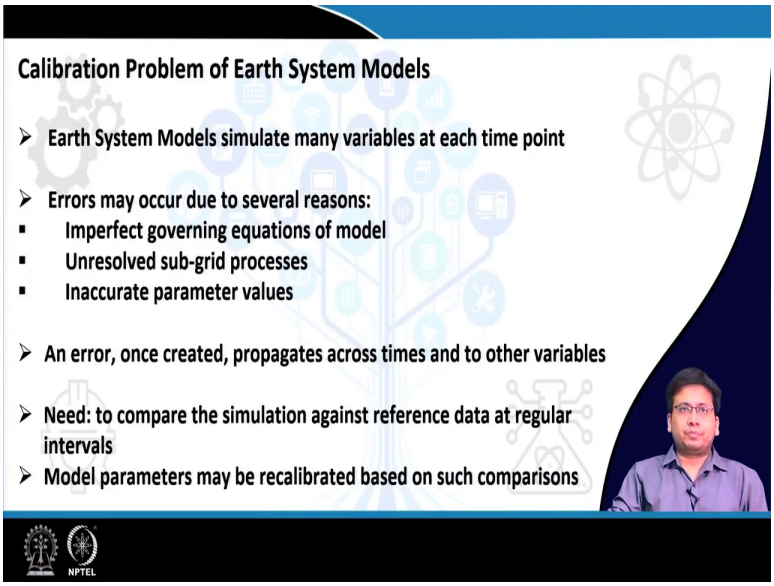
Hello everyone, welcome to lecture 39 of this course on Machine Learning for Earth System Science. We are currently in module 5, where we are dealing with Machine earning for Earth System Modelling and the topic of this lecture is Data Assimilation for Earth System Model Correction.

(Refer Slide Time: 00:42)



The topics which we are going to cover today is first we will understand what is the issue of bias in process-based models for I mean process based earth system models then we will discuss how or what are the sources in which these models can be calibrated at run time to get rid of these biases and thirdly how machine learning can be used in different ways to achieve this purpose.

(Refer Slide Time: 01:08)



So, first of all let us discuss about what is the main problem we are trying to address here. So, we have discussed at length about earth system models and now earth system models they try to simulate very many variables which are of their relevant interest and these variables are simulated over a period of time. So, at every point of time the model simulates some values of all those variables.

Now these models or sometimes make errors or that is to say at the end of the simulation the values of the different variables are different from what should be like according to observations and such errors can occur due to several reasons. First of all the governing equations of the models may not be perfect.

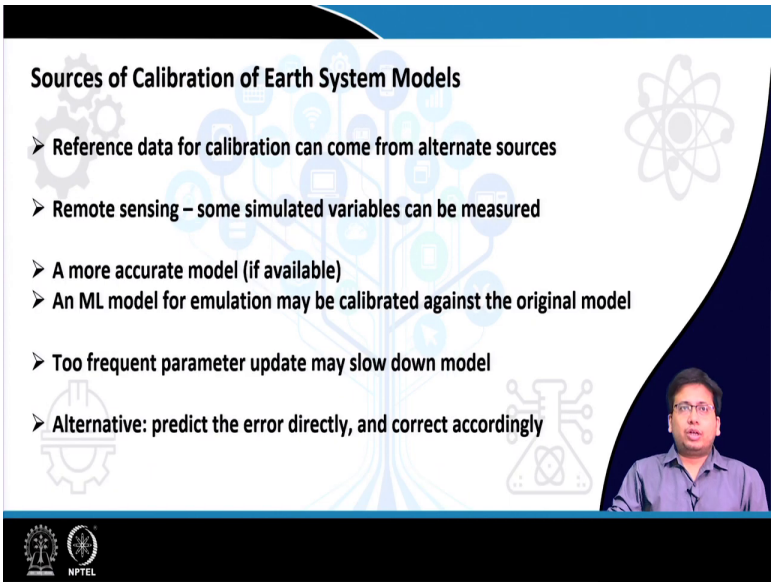
They may be like it is possible that the model uses a first order or linear equations while maybe more complex equations could have been more suitable or maybe vice versa or it may be that the some of the parameters involved in those equations their values are not known properly and it can also be that the various subgrid processes are unresolved.

This last issue we have already discussed at length in the previous lecture, but due to all these issues the final result of the even the intermediate results of these simulations are often off the mark. Now also if an error is created sometime in the middle of the simulation at an intermediate

step, this is likely to propagate across times and to other variables and at every subsequent time point larger error will be incurred.

And hence the final by the time the simulation ends like the a lot of error might have been incurred and as a result we may get a very wrong or inaccurate output. So, what is necessary is to compare the simulation again some kind of reference data at regular intervals and recalibrate the model accordingly.

(Refer Slide Time: 03:17)



Now, where does this kind of reference data come from? It can come from a very or like different kinds of sources and depending on what is the type of or what exactly the process model is doing and also how we incorporate that data into the model that also can be achieved in different ways.

First of all like we may have some actual observations of those variables which are being simulated especially in the historical period, we may have some observations say through remote sensing. So, if that is the case then the simulated value and observed values can be compared and any error can be like reported to the model so that the model can update itself in whatever way suitable.

Like in case a more accurate model is available. In that case that accurate model can like provide the this feedback to the model which is of our interest. For example, we have already discussed

several times how machine learning models can be used for emulating the process based models. So, in this case we can assume that the process based model is perfect and the machine learning model is trying to like emulate it.

So, whenever the machine learning model makes any errors such error can be compared against that I mean the spread the error can be computed by comparing the predicted value against the what is the value which is computed by the main process model and the error can be communicated to the machine learning model so that it can like update its parameters accordingly.

And or even if it does not update it is parameter at least reset the values that it has predicted so that in the subsequent time steps it does not keep on incurring the error. The problem with regular parameter estimation is that like parameter estimation of models is often like is often a time consuming step especially if the if we are talking about some kind of a deep learning model which has a typically millions of parameters.

So, we know that like a training a deep learning model with means like optimizing all those parameters which can easily take a lot of time. Now, if we are now even if we do not update all the parameters even if we are updating only a subset of the parameters even that may not be acceptable because we are we want to run these models in real time. So, like stopping to update every now and then is going to make the system inefficient.

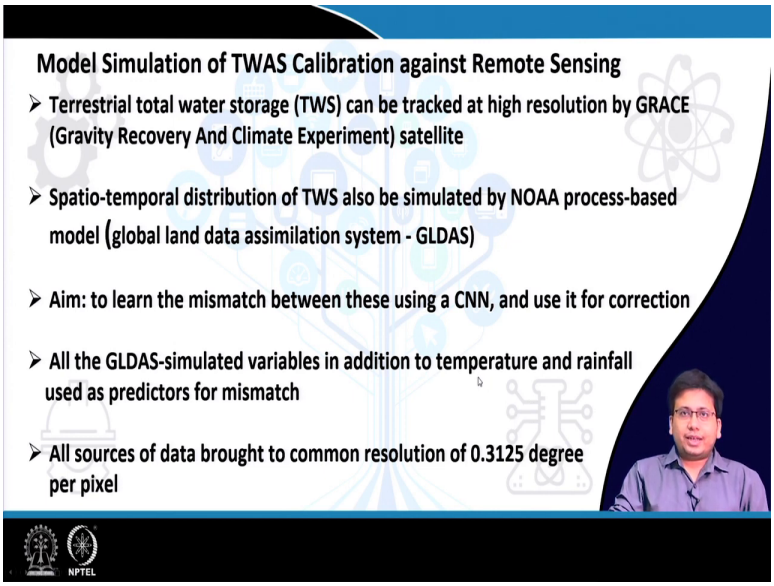
So, the alternative might be to like instead of like actually calculating the or actually recalibrating all the parameters just somehow be able to predict the error at regular intervals of time and incorporate the those errors so that you so that the we make corrected predictions and in the subsequent steps we use those corrected predictions for continuing the simulations. In this case even if the model never becomes perfect still we have a corrective mechanism in place.

(Refer Slide Time: 06:34)

So, a paper which like uses this technique is the this paper; Combining Physically-Based Modeling and Deep Learning for Fusing GRACE Satellite Data: Can We Learn From Mismatch? So, here basically the aim is to simulate the total water storage in the ground. So, this can be simulated by certain process-based models which have their imperfections, but the total water storage can also be measured through proxy variables using the GRACE satellite.

We had already discussed this issue at a using the NDVI and so on in an earlier lecture. So, the task here is like treat those satellite observations as ground truth and when whatever value is being simulated by the process models calculate at every step or at regular intervals the mismatch between the simulated values and the satellite observed values and just report that mismatch so that an action can be taken.

(Refer Slide Time: 07:36)



If the action in this case is not the re or need not be the re updation of all the parameters of the process based models, but the simply the correction that is addition of the correction. So, the like the terrestrial total water storage or TWS, this can be tracked at high resolution by the GRACE satellite.

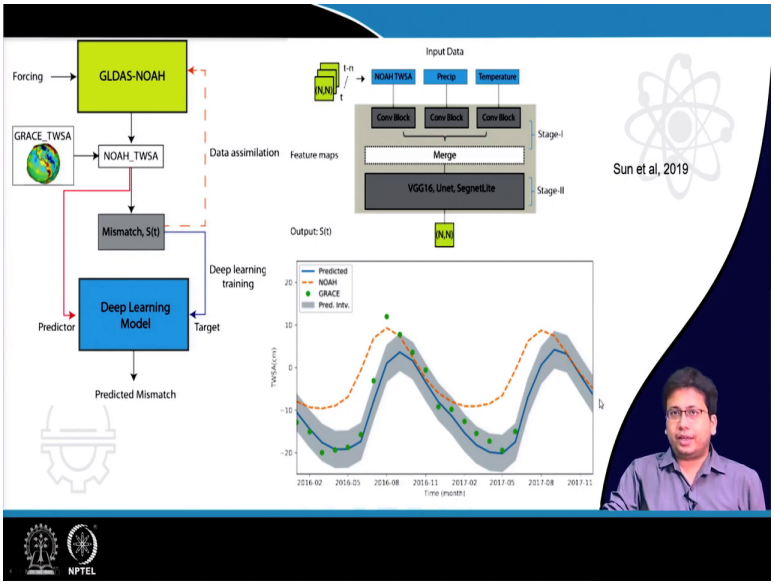
The spatio-temporal distribution of TWS is also simulated by the process-based model called the Global Land Data Assimilation system or GLDAS this is developed by NOAA which is of course, a very famous agency in based in US. The aim here is to learn the mismatch between these two sources using a CNN and use it for the correction.

So, the correction of the GLDAS simulation, the satellite is assumed to be the ground truth. Even though it need not be, but in this case we assume that like we our aim is to simply calibrate the this simulated values with what is observed or what is estimated using the satellite. Now, all the GLDAS simulated variables in addition to the temperature and rainfall are used as predictors for mismatch.

And all be for that purpose all the sources of data are brought to a common resolution. So, the this GLDAS the process based model it carries out it is simulation at a particular resolution, the satellite observations will be at a different resolution. So, it is necessary to first of all bring them

to a common resolution and in that uniform resolution at every pixel what they intend to do is they intend to predict what how much mismatch there is going to be.

(Refer Slide Time: 09:25)



So, this is in like as you can see in this figure, this is the what is the simulation by the GLDAS model and this is the observations from the GRACE satellite. So, these two are compared and the mismatch is calculated. So, now, a model is trained whose aim is to who like which basically takes as input the simulation results and it tries to match the simulation results plus some predictors of error like precipitation and temperature to predict how much error is going to be.

So, at any given like once this model has been trained at any given point of time like based on the whatever values of groundwater has been simulated by this model, the neural network will be able to predict how much error it has incurred at which at each pixel level provided it is also equipped with these additional variables like precipitation and temperature.

Now, why these two predictors are chosen? Like the authors have somehow realized that these two variables or covariates are important in or the these two somehow drive the errors which are in the like in the process-based models. May be the process-based model is not able to take these two parameters very or I mean these two variables in a very accurate way and as a result of that they tend to incur errors.

There are of course, may be more predictors or more covariate variables which contribute to such error in the modeling but in this paper the authors have focused on these two. So, the deep learning model which they are used using in this case it takes into account it takes as input the full spatial maps of the groundwater as simulated by the process based model at different point of time.

So, it is a sequence of spatial maps these are the input and the precipitation and temperature as already mentioned these are also the different source of inputs. So, like it is basically something like a convolutional neural network initially. So, like so, all these things are subjected to convolution followed like because these are spatial maps.

Of course, they will have to be treated with a CNN. Like in this case because it is a sequence of in maps maybe like something like an LSTM or CONV LSTM may also have been used, but the authors have not used it for whatever reasons. Now, all these things are merged together and then fed into this kind of a model for image segmentation.

This VGG16, Unet, Segnets like as you know the like we have discussed earlier these are basically used for image segmentation and the output of this is again another spatial map of the same resolution as the input and here each pixel contains the predicted error at that pixel level.

Now this error map is then provided as feedback to the input which is basically the model simulations so that the model simulations can be corrected by adding the error values to them and then these model, so, the model equations parameters etcetera remain are unchanged, but the simulated values are updated so that the simulation can proceed from those updated values.

So, just this, look at this cartoon here. So, this dashed line is let us say this is how the process model, it simulates the total the ground water over a particular region. So, let us say in this case we are just considering the mean ground water level across the world or across a particular region. So, this is what it looks like.

As you can see the variance in this case is relatively less, but compared to the observations which are these green dots. So, you can see that in case of the green dots the observation is clearly higher. So, this is again in line with something which I had told in the previous lectures also that when process-based models are trying to simulate something. They are often able to get the mean

correct, but the variance the variability they tend to underestimate. Here also you are seeing that happening.

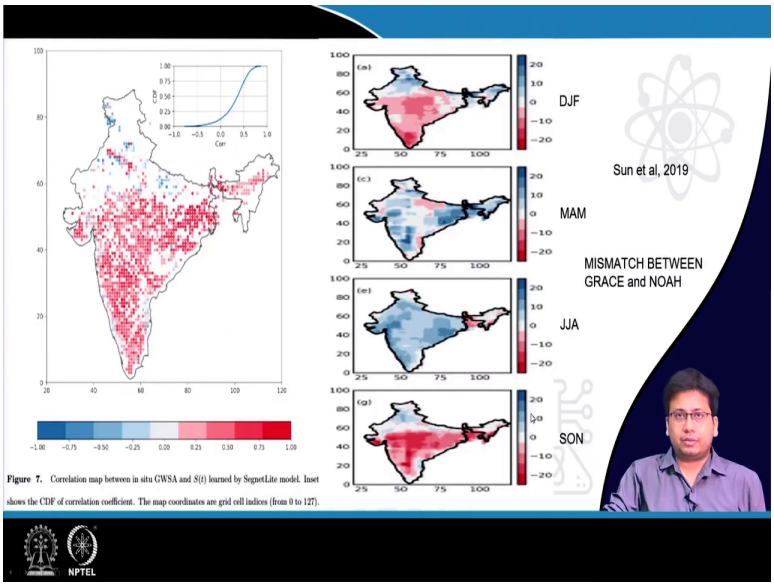
Now when they like merge these two sources or they use this kind of correction based on neural network then what they get is, they get the this blue plot and as you can see this blue plot is much closer to the observations. Now, this is done not only for the trade like let us say that training has happened till this point of time, for this is the future and for which there are no observations.

But you can see that in the this future also the model the corrected model is continued to continuing to predict same as or like it seems that the predictions by the corrected predictions are significantly different from what the model would have simulated had the correction not taken place. Of course, this will not happen for long because like in the absence of errors soon or later the errors the model by errors will again creep in.

So, we see that for from this point onwards again the these two approaches are converging that is because like the this error this scheme is not altering the model itself which is only like nudging the values back to what to the correct values and that is not I basically not allowing the errors to accumulate, but this effect can stay like once of course the training period ends, once we go into the future further observations will not be available from the remote sensing.

So, further corrections will not be possible and hence after a point of time the model will start making errors again. The idea is can we maximize the period for which the model can be can run error free. So, we can see that from here to here for about one year or in fact, not one year for a few months, the model is able to run error free before the errors start creeping back in.

(Refer Slide Time: 16:09)



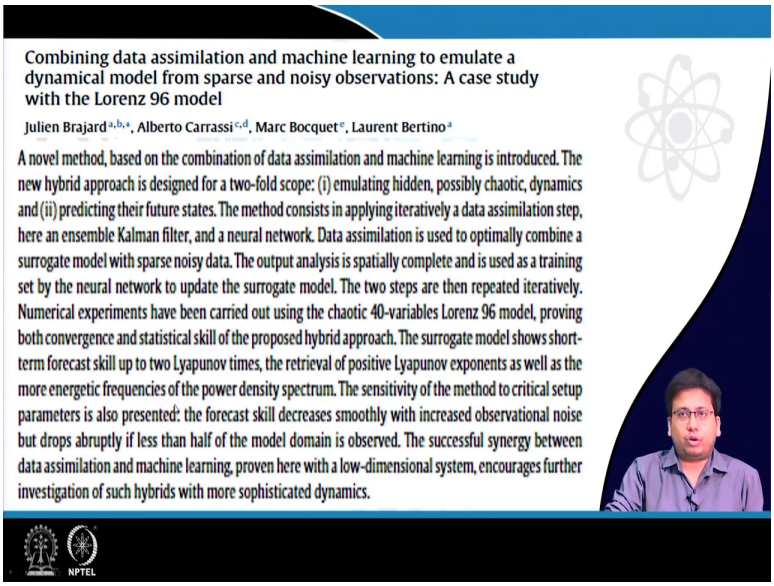
So, this is how the we see the results. So, like if we compare the model simulations against the in-situ observations. So, note that the previous observation the model corrections had been done on the basis of GRACE satellite estimates, but GRACE satellite estimates may themselves have certain biases that were like that is the topic of another paper which we had discussed earlier, where the like the GRACE satellite observations were somehow calibrated against the in-situ observations.

So, now here the process model has been calibrated against the GRACE satellite observations, but how about the actual in situ observations? Is it able to do like is the model is the process model after correction able to match the in-situ observations? So, it turns out that to a large extent yes in some places the error especially in the North India the error is near 0, though in other parts of India there is a slight over estimation of the ground water level.

Like so, I am sorry that is wrong. In fact, this is actually the correlation coefficient map. So, we see that in like various parts of North India the correlations are a bit low that is closer to 0, but in large parts of Peninsular India the correlation is quite high. In fact, close to 1 which means fine agreement between the in-situ observations and what is simulated by the model.

And like in the different month-wise also this kind of the error has of prediction of this ground water has been studied and here we see that in certain months like the there is seems to be some a bit of overestimation and in some other months it is found to be underestimation though especially in the middle month in the May to August period the such errors are relatively low.

(Refer Slide Time: 18:10)



Next we consider another paper. So, in this case like we had a process-based model which we were calibrating against the remote sensing observations. In this case let us consider a situation where we have a process-based model that we are trying to emulate using a neural network and we are trying to like improve upon that. So, for this the process based model which we study in this case is actually is not an actual process based model, but only a conceptual model, they called the Lorenz 96.

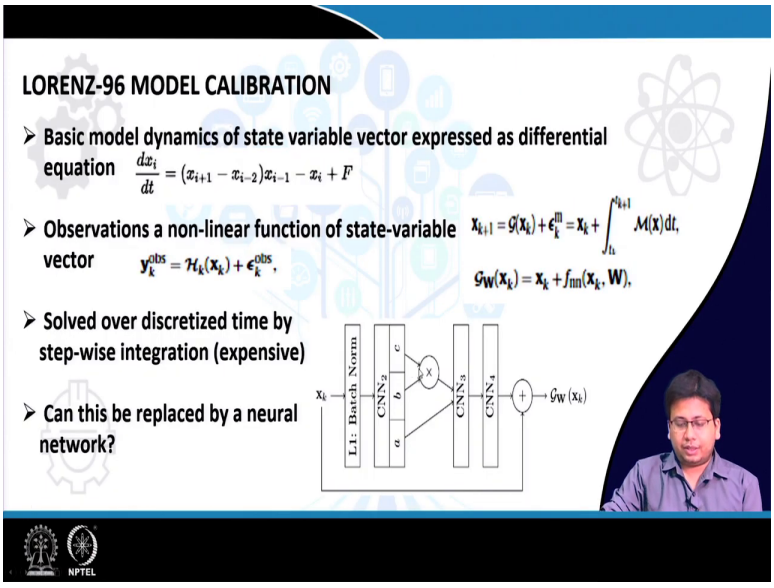
So, this Lorenz 96 is a well known model which is that is it is used like it can be considered as a prototype for various process based model even though this model itself is quite straightforward. The novel model a novel method, based on the combination of data assimilation and machine learning is introduced. The new hybrid approach is designed for a two-fold scope that is emulating hidden or possibly chaotic, dynamics and secondly, predicting their future states.

So, this Lorenz 96 model is basically a dynamical system which is used for predict like which is used for predicting or which can be used for chaotic processes. The method consists in applying iteratively a data assimilation step, here an ensemble Kalman filter and a neural network. This data assimilation is used to combine optimally combine a surrogate model with sparse noisy data.

The output analysis is spatially complete and is used as a training set by neural network to update the surrogate model. The two steps are then repeated iteratively. Numerical experiments have been carried out using the chaotic 40-variable Lorenz 96 model proving both convergence and statistical skill of the proposed hybrid approach.

The surrogate model shows short-term forecast skill up to two Lyapunov times, the retrieval of positive Lyapunov exponents as well as the more energetic frequencies of the power density spectrum. Do not worry if you do not understand all these concepts. So, these are related to non-linear dynamics and this part is not relevant to what we are discussing.

(Refer Slide Time: 20:33)



So, let us focus on only the first part. So, this is the Lorenz 96 model. So, like as you can understand this is a differential equation. There are multiple variables. So, like here they mentioned 40 variables. So, these i they take values from 1 to 40. So, each the derivative of each

variable is expressed as a function of the other variables and that variable itself and there then there is an external fact called thing called forcing.

So, like this Lorenz 96 model proceeds by like at every point of time this derivative is calculated for each variable and accordingly every variable is updated. So, calculating these derivatives and updating them at like very at like very short intervals of time, I mean infinitesimally short intervals of time that is of course, potentially a time consuming step.

And furthermore at every point of time we have observations from a like each of those variables, here denoted by k. So, there is a non-linear operator H, which like which might be different for each of those 40 variables and or it can be the same also. For the sake of generality let us just assume they are different and then there are some noises also. So, basically the observations we get from the Lorenz 96 model are these things, these $y's$.

And then the problem is using the y you have to predict the like estimate that the x at that point of time and predict the following future values of x and hence y. So, as I said the solving this differential equation at every point of time and proceeding accordingly is the, it can be quite slow. So, a possibility is like.

So, this is how it the whole thing works that is like we discretize the time into infinitesimally small intervals and like we carry out this integration over those small over those for small intervals this integrand $M(x)$. So, this is basically a representation of the system dynamics and then like.

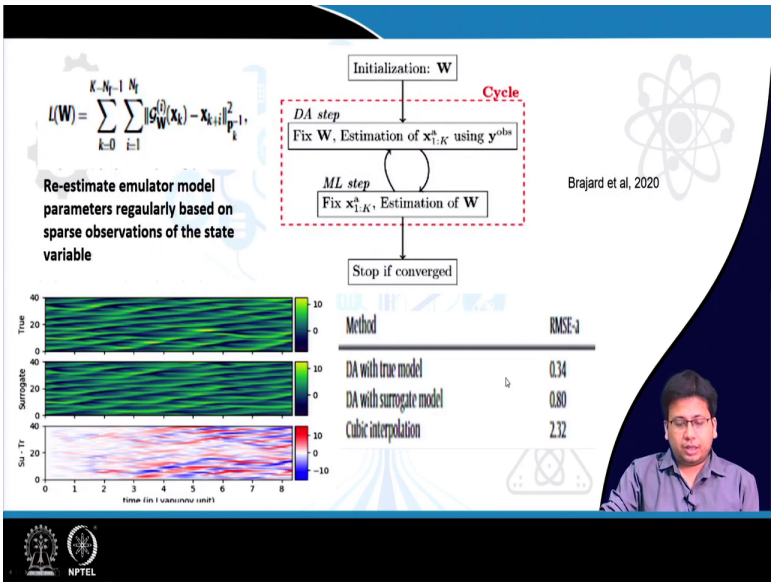
So, this basically gives us the change of that variable over that small interval of time and accordingly the x is updated. The alternative is like this integration like can it be replaced by a neural network. That is basically what we are saying is we are will suggest this like we are proposing to use a neural network to emulate this integration process.

So, the like for that purpose the authors have proposed this kind of a neural network whose like input will be x_k at any given point of time and it is output will be the this $G(x_k)$ that is basically the x_k plus the whatever is this part ok. So, the x_k plus that thing is of course, handled by this kind of a self-connection.

I mean this is a pretty obvious step. We like even if this was not there and the x_k was added separately then also the neural network would have been fine, but the. So, basically like what this neural network is trained by in it is trained in the initially before the model is operational by just mapping the input variable at every point of time that is the its initial value at time t and the predict and it should predict the next value at time point $t + 1$.

So, that is how it this model has been trained. Now because it is a 40 this observation these are as I mentioned these are 40 dimensional and that these dimensions also interact with each other in a like a non trivial way. So, that is why they have taken the CNNs so as to like help in the these things. So, especially like instead of it being a vector this could have been more complex things like matrices also. In that case it would have the CNN should have made even further sense.

(Refer Slide Time: 24:53)



So, this is like the approach of using a that is emulator for the Lorenz 96 model itself, but now we are telling something more than this. We are talking about data assimilation. So, that is to say we need to like the as more and more data comes in or more as more and more observations comes in can we actually keep on updating the neural network or the and its parameters, in this case W. So, the idea is as follows.

Using the like once you like initially you have trained the neural network, you have got the W which is basically the all the weights the all the parameters of the neural network. So, using the W you can predict the next few steps using the observations you have obtained so far. But those predictions may go wrong over time due to like various factors. Maybe your estimation is not correct or whatever.

I mean if we have a neural network of course, it will be making some kind of errors because the emulation will never be perfect. Now, the idea is that once we have the observations, now keeping those or like keeping these your predictions as like pinned down to the observations as they come like you reestimate the values of W the parameters and you keep on repeating this process until it has converged.

So, like you define a new loss function that is the these like you will be receiving the updated values at sparse time intervals, not regularly. At every point of time you will not be getting, but the whole period of simulation you just divide into certain large intervals and maybe at the end of every interval you will get an observation.

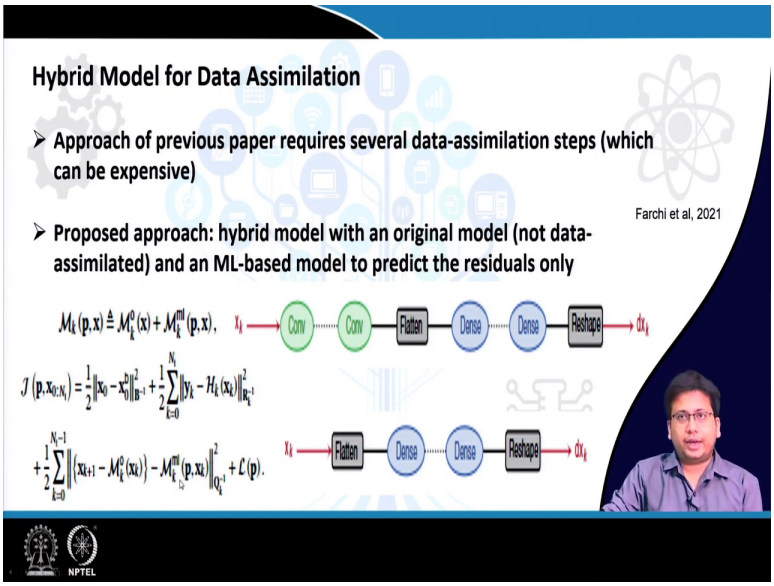
So, basically you define the loss function where the value which is simulated by the or which is predicted by the neural network is compared to the that observation and this process is like is used and this or this loss function you regularly utilize it to recalibrate the neural network so that it all it is parameters keep on updating.

And it is found out through experiments that if you do this for a long enough period of time, then your neural at beyond a particular point further updates will not be needed and your neural network will be able to emulate the Lorenz 96 dynamics with more or less high accuracy.

So, this is these are actually the simulations of the Lorenz by the Lorenz 96 model. So, like along the y-axis you are seeing those 40 variables and along the x-axis there is the time. So, at every point of time you have the values of those all those 40 variables and their values are like color coded.

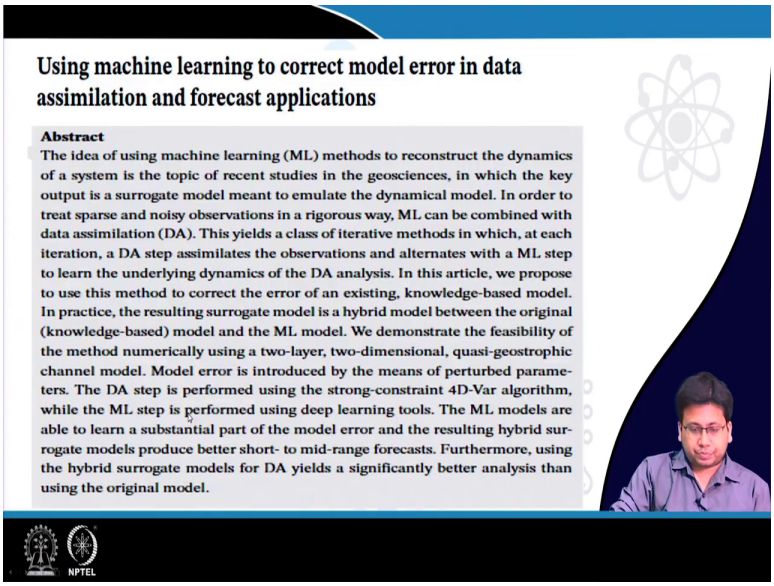
So, this is these are this is the actual result we get using the Lorenz 96 model itself and this is the result which we get from the surrogate model which uses the neural network for predicting the next step and as you can see that the errors are reasonably low, it is not very high.

(Refer Slide Time: 28:24)



And the now this particular approach has one drawback that it has to be that is the neural network has to be recalibrated several times at least in the initial stages till so that is *W* values are updated as already mentioned and now this updation of this neural network is a time consuming step. So, if we want to do the emulation in real time we cannot really hope to spend so much time in this matter.

(Refer Slide Time: 28:56)



So, an alternative has been proposed in this paper. It is basically a very similar problem, but the idea is you do not update every like at every regular intervals or you do not update the whole thing at the same time. Instead, the this prediction of the next value is not done entirely using the neural network there is a actually a base model something which they call as a knowledge base or original model which does not take this data assimilation into account.

It like let us say it is some kind of a model which has been trained initially before the that is it is like it does not need to be like or we do not aim to update this model in real time. So, it is predictions may not be perfect, but it will give at least some kind of a baseline. And on top of that the extra error we will try to correct using the machine learning model which will be updated in real time.

But the even this machine learning model will not we will not update all the its parameters at every step, but like with even sparse intervals we will be updating this machine learning model. So, this is the loss function of this machine learning model part. So, like the like as you can see here this is like the difference between the actual observations y_k and like what is the observation according to the model. So, x_k is the like estimated system state.

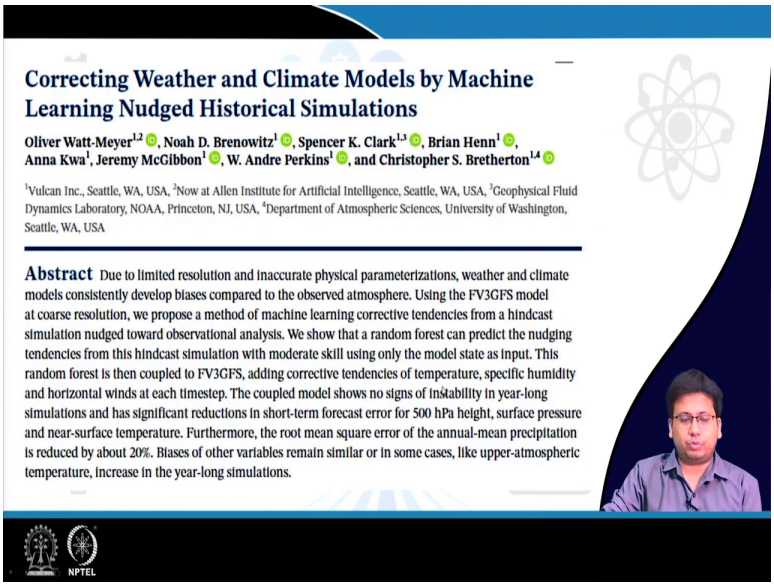
So, according to x_k, $H_k(x_k)$ should be the model should be the observation, but the observation is y_k. So, what is the difference between that? So, that is one source of error. Then like in the with the original model with according to the original model also x_0 let us say is or let us say x_0 is the like observed states. The states themselves may be observed at regular at sparse intervals.

So, that has to be compared against the like what is estimated by this base model and then finally, this residual how well that is like simulated with the help of the neural network that we are developing. So, there also that is another source of error. So, the at every that is at regular intervals of time this loss function is calculated and the parameters of this neural network that is the this only this residual part is updated using the like so as to minimize this complex loss function.

And the thing is in this case when even when this neural network is being updated that does not hamper the prediction because we already have this original model. So, even when the neural network is being updated the predictions can go on using the using this original model.

And once the neural network is updated then the like it can be used to correct the further observations. And so, in this case instead of having a like this architecture they are considered two different architectures of neural network and they are found to be working better.

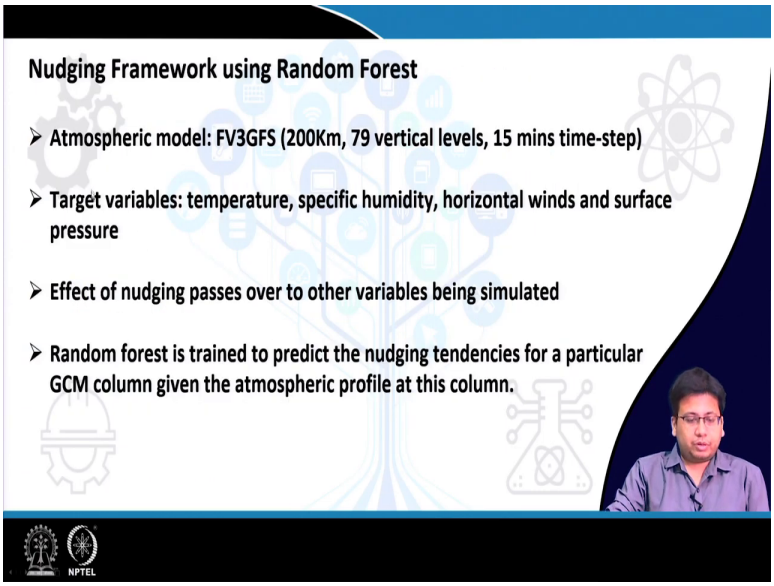
(Refer Slide Time: 32:17)



And lastly we come to another approach where random forests are used to for what is known as nudged historical simulation. This is actually somewhat similar to the first paper that we discussed. So, here due to limited resolution and inaccurate physical parameterizations, weather and climate models consistently develop biases compared to the observed atmosphere using the FV3GFS model which is a like an atmosphere atmospheric process model.

At course resolution we propose a method of machine learning corrective tendencies from a hindcast simulation nudge towards observational analysis. We show that a random forest model can predict the nudging tendencies from this hindcast simulation with moderate skill using only the model state as input.

(Refer Slide Time: 33:11)

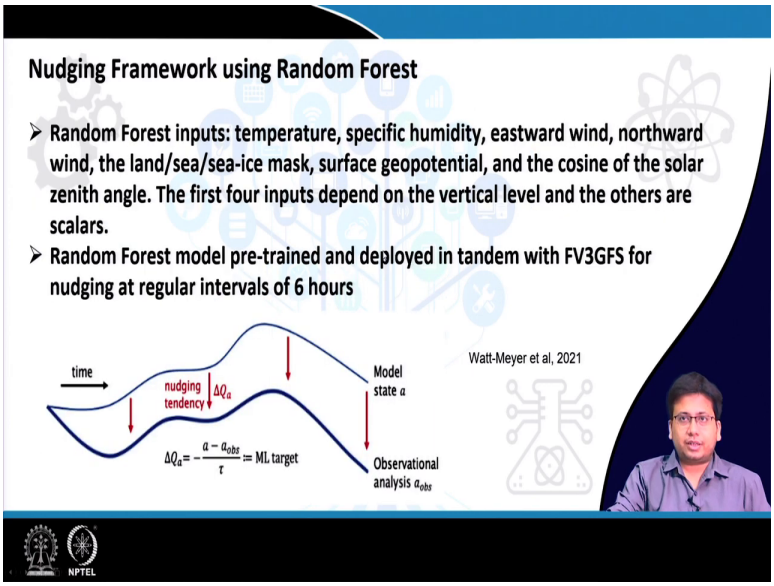


So, like in this case the this is the atmospheric model which like it works at 200 kilometer resolution which is roughly 2 degree by 2 degree. It has 79 vertical levels and the time steps are 15 minutes each. The target variables which are going to be nudged or that is to say compared against some observations or something and updated; these are the temperature, specific humidity, horizontal winds and surface pressure.

Now although we will specifically correct these variables at regular intervals of times, but the effect of their corrections will pass on to other variables also which are linked to these variables through the governing equations of the model. So, the a random forest is a random forest model is trained to predict the nudging tendencies for a particular GCM column given the atmospheric profile at this column.

That is like basically by column what they mean is like the in case of a GCM the whole system is divided into columns of like at a like we say that at every location, there is a like a vertical column of the different layers of the atmosphere. So, note that 79 vertical levels are being considered.

(Refer Slide Time: 34:30)



So, at the like at every at each of those levels it is necessary to recalculate the losses. So, the random forest takes as input the temperature, specific humidity, eastward wind, northward wind and all these variables as their inputs. So, some of these inputs are specific to the 79 to each of the 79 vertical levels and the others are just scalars.

Now, what the random forest model predicts is the amount of error in each of these four target variables. There are there is its input is all the values of all these variables at for every column and their output is the possible error that the atmospheric model has incurred. So, how does it measure that error? By comparing with observations. So, this is how the model is going and this is how the what the observations are.

So, at every point of time we see these we note these errors. These errors are actually in subsequent time like. So, this is like the in the training phase we have of the we have the model simulations as well as the observations. So, there is a random forest model which learns to map the model state to the errors.

(Refer Slide Time: 35:46)

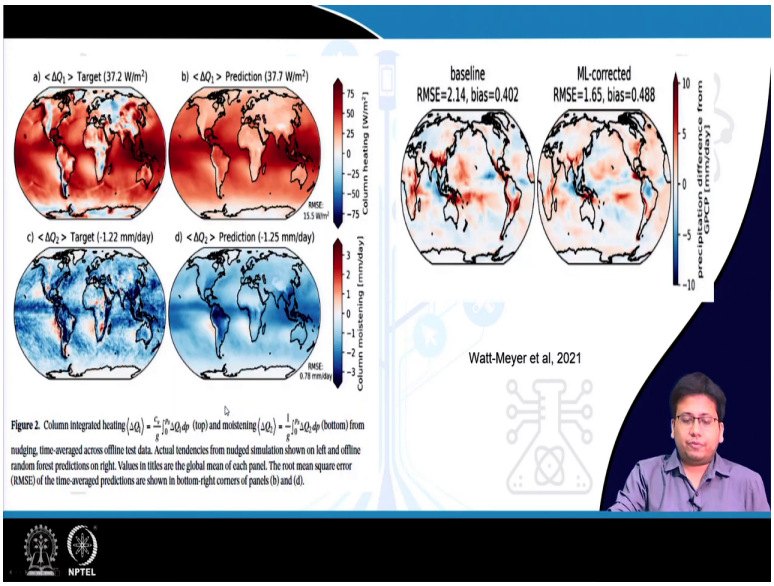


Figure 2. Column integrated heating $\langle \Delta Q_1 \rangle = \frac{c_p}{g}\int_0^{p_s} \Delta Q_1 \, dp$ (top) and moistening $\langle \Delta Q_2 \rangle = \frac{1}{g}\int_0^{p_s} \Delta Q_2 \, dp$ (bottom) from nudging, time-averaged across offline test data. Actual tendencies from nudged simulation shown on left and offline random forest predictions on right. Values in titles are the global mean of each panel. The root mean square error (RMSE) of the time-averaged predictions are shown in bottom-right corners of panels (b) and (d).

And then in the furthest situations where there are no observations to guide I mean in the future, in that case this random forest model itself can predict the amount of error and hence nudge the model to like recalibrate its simulated variables according to that error so that is to say so as to minimize those error.

And it turns out that it works quite well also that is there is a like in compared to baseline that is if you run the atmospheric model without any kind of correction it incurs certain RMSE, but if you do the ML correction then you get a significant reduction of the RMSE even though there might be a slight increase in bias. And this is like this can also be studied for the individual variables that we are trying to nudge.

(Refer Slide Time: 36:35)

So, these are the references to the four different papers that we discussed today. So, that brings us to the end of this lecture number 39. So, in the next lecture which will be the last one we will see some further applications of machine learning models especially for understanding climate change and so, till then bye.