Referee's comment:

Thank you again for thoroughly responding to the reviewers' comments: At this point, the manuscript is clear and the authors' framework reproducible. However, the manuscript's first key point remains to be proven because (1) UNETx and U-STNx are not compared in a systematic fashion (1.1); and (2) U-STN is still misleadingly presented as equivariance-preserving in parts of the manuscript (1.2). Additionally, WeatherBench models may be unfairly compared to a version of U-STN including data assimilation cycles (1.3), and U-STN is still framed as a physics-informed machine learning (ML) framework (1.4). Finally, there may still be discrepancies between the shared code and what is reported in the manuscript (2.1). I continue to think that this manuscript will be a welcome contribution to GMD once revised so as to not mislead readers about equivariance preservation. Overall, it would be helpful to clarify whether the improved test performance of U-STNx reported by the authors without data assimilation (e.g., Fig 6) is truly the result of some form of equivariance preservation, or whether using a bottleneck and a spatial transformer simply prevents overfitting the WeatherBench training set.

Authors' response:

We thank the referee for their insightful comments which have substantially improved the quality and accessibility of the manuscript. Herein, we address each of the referee's comments. All the changes in the revised manuscript have been tracked in blue (note that the tracked manuscript tracks all changes from the original submission) until round 2 and the last round of revision has been tracked in red.

We want to emphasize upfront regarding the major issue that the referee has been pointing out about the equivariance-preserving nature of the network: we agree with the referee that having an equivariance-preserving latent space via a spatial transformer is not the same as a fully equivariant network. Hence, we have decided to remove *"equivariance-preserving"* throughout the manuscript. Instead, whenever we have talked about the advantages of the spatial transformer, we have explained that it *may help* in capturing rotational and scaling features only in the transformation in the latent space where it has been implemented. We hope that this would avoid any confusion amongst readers that our network is indeed *not fully equivariant end-to-end*.

Referee's comment:

1.1 Comparison between U-NET and U-STN

The first key point of the manuscript relies on comparing U-STN with U-NET to establish U-STN's improved performance compared to U-NET ("improving the forecast skill by 45%"). This requires being as precise as possible when comparing
U-STN and U-NET:

- *According to Table 2, U-STN has three additional layers (10/11/12) of 200/100/50 neurons. Would it be possible to transparently list the number of trainable parameters (e.g., the output of model.summary()) for both the U-STN and U-NET? If they are different, what would suggest that this is a fair comparison?*

- *For example, how is it possible to know whether U-STN improved performance over the test set comes from the spatial transformer or from the bottleneck created by layers 10/11/12. Would it be possible to introduce a U-NET with the exact same architecture as U-STN (minus the spatial transformer) to test whether the improved performance simply comes from not overfitting the WeatherBench training set?*

- *"the optimal set of hyperparameters that have been obtained after extensive trial and error": This is a vague statement in the age of formal hyperparameter optimization (e.g., Hyperas, Hyperopt, the Keras tuner, SHERPA, etc.) that does not establish in any way that the architecture of U-STN is comparable to that of U-NET. Would it be possible to:*
  - List the hyperparameters and their considered range during the manual search?
  - Systematically report the performance of U-NET and U-STN over the training/validation/test set (as opposed to
  just the test set) to identify whether U-STN is a better fit, whether it generalizes better, etc.?
  - This could also help better understand the role of the spatial transformer (e.g., does it help because it prevents
  overfitting the training set?) as equivariance is not generally preserved by the U-STN framework.

Authors' response:
We thank the referee for this insightful comment about the relative complexity of U-STN and U-NET and how that affects the prediction horizon of each of these models. We would first like to point out that the U-STN and U-NET have **separately** undergone hyperparameter optimization (albeit via trial and error, nonetheless, an extensive one). This is to say, that a U-NET with more depth, more width (large number of filters), or larger/smaller kernel sizes would not have better prediction horizon than the current U-NET. So, comparing the number of parameters in U-STN and U-NET separately does not lead to any conclusion about their relative performance. Still, based on the referee's comment, we have added the model.summary() command in our codes to transparently report the total number of trainable parameters in each of the models (U-STN: 2,461,965, U-NET: 291,777). Of course, this discrepancy in the number of parameters is largely due to the dense layers in the U-STN. Further, herein, we include a few of the tables that we have obtained from our hyperparameter tuning experiments to show that larger number of parameters in the U-NET would not lead to better prediction horizons in Table 1 of the paper.

- A larger number of parameters in the U-NET as compared to the U-STN would not lead to a better model (longer prediction horizon) since the depth, width, and kernel sizes have been optimized for this U-NET through hyperparameter tuning. It must be noted, that owing to a limited size of the training set, an overly complex model may not necessarily lead to better performance. We report some of the results on hyperparameter tuning in Table 1.

- We have performed experiments with adding different number of dense layers to the U-NET when performing hyperparameter tuning. We show the prediction horizons on the validation set in Table 2. As is evident from Table 2, adding these dense layers in the U-

NET does not lead to any improvement and in most cases deteriorates the performance of the U-NET.

Table 1, and Table 2 clearly shows that simply increasing the number of parameters in the U-NET would not necessarily lead to better prediction horizons. While we have done further hyperparameter tuning, these two tables summarize that neither including the dense layers (as in U-STN) or increasing the number of filters lead to better prediction horizon. Based on significant trial and error, we believe that U-STN outperforms an optimal U-NET in this problem. We hope that the referee finds these hyperparameter tuning logs sufficient to show that the spatial transformer layer does have an added advantage as compared to regular U-NET.

Table 1. Hyperparameter tuning for number of filters in each layer and kernel sizes on the prediction horizon of U-NET on the validation dataset. Note that increasing number of filters increases the number of trainable parameters. **Note that the last row of this table (red) has 13,773,665 trainable parameters which is more than that of U-STN but does not show improvements in performance.** The highlighted element in the table is the optimal number of filters in each layer with the optimal kernel size.

Number of Filters	Kernel Sizes				
	3x3	**5x5**	**7x7**	**11x11**	
16	~77	~77	~75	~75	Prediction Horizon (hrs)
32	~88	~96	~93.3	~91	
64	~82	~95.3	~92	~81	
128	~88	~95	~88	~88	
256	~74	~74	~72.2	~68	

Table 2. Hyperparameter tuning for number of filters and dense layers before the bottleneck on the prediction horizon of U-NET on the validation dataset. Here as well, we can see that the optimal performance of the U-NET can be obtained without any dense layers in between. Note that the last row of this table has two orders of magnitude more parameters than the optimal U-STN.

Number of Filters	No dense layers	3 Dense layers, 200-100-50	4 Dense layers, 300-200-100-50	5 Dense layers, 400-300-200-100-50	
16	~77	~62	~62	~61.5	Prediction Horizon (hrs)
32	~96	~78	~72	~67	
64	~95.3	~72	~69	~67	
128	~95	~72	~68	~68	
256	~74	~72	~68	~66	

Referee's comment:

1.2 U-STN is not equivariance-preserving

U-STN is still framed as a framework preserving equivariance:
[L4 in the abstract] "to preserve a property called equivariance"
[L44] "based on building physical properties called equivariance"
[L102] "Introducing the equivariance-preserving"
[L338] "to preserve equivariances".
While the authors added a discussion at the end of subsection 3.1.2, the above statements would mislead most readers (especially those focusing on the abstract/introduction/conclusion) into thinking that this framework ensures equivariance preservation. Assuming that the above claims relate to SO (3) equivariance, which if I am not mistaken would here be defined along the lines of:

$$\forall\, \theta, \forall Inputs, U - STN[T_\theta(inputs) = \tau_\theta(U - STN(inputs))$$

would it be possible to clarify that:

- *The framework is not equivariant because it learns one set of parameters per sample, and that these parameters are not physically interpretable according to the authors (in their response to reviews),*
- *The U-STN architecture includes skip-connections, which prevent equivariance preservation at multiple levels of the architecture (up6 and up7 in the code use conv2 and conv1).*

Note that other manuscripts using equivariance preservation (e.g., [1]) typically define equivariance mathematically and exactly stipulate whether the property is satisfied or not upfront.

Authors' response:

We agree with the referee that using the term, *"equivariance-preserving"* can be misleading to the audience. Hence, we have decided to remove this term whenever referencing our network within the revised manuscript. Instead, whenever we talk about the advantages of the spatial transformer, we would explain that *it may help in capturing rotational and stretching features* in the transformation of the latent space. We have further added Lines 153-154 explicitly mentioning that the entire network is not equivariant by construction. We hope that this change would remove all confusion regarding the network's properties amongst the readers. We thank the referee for pointing this out and having the invigorating discussion.

Referee's comment:

1.3 Misleading comparison with WeatherBench

[Table3, Sec 6.2]
1. Does the U-STN12 compared to WeatherBench models assimilate data from the test set? In the affirmative, I
recommend comparing WeatherBench models to the version of U-STN12 that does not use data assimilation (e.g., the version presented in Fig 6) and clarifying the text accordingly.

2. Are the authors using the same test set (their paper uses 2018) as the test set used to report the WeatherBench models' performance (2017 and 2018 according to [2])? This would also prevent a fair comparison and I recommend using the same test set as in WeatherBench while explicitly acknowledging that samples from the validation set will be used for this particular comparison.

Authors' response:
We thank the referee for raising this point. When comparing with WeatherBench we have used U-STN12 **without DA.** We have indicated that in the title of section 4.1 as well as in the appendix in section 6.2 (Line 390).

We further want to emphasize that the prediction of U-STN12 or U-NET12 will not change with the choice of the year in the testing dataset (we have tested on data from 2017, 2016, and 2015 as well; for these years the training samples would be reduced as well). We have reported prediction horizons across multiple initial conditions sampled from the testing dataset.

Based on the referee's suggestion, we include Table 3 showing the prediction horizon over 30 initial conditions, sampled from years 2017, 2016, and 2015.

Table 3. Prediction horizon of U-STN12 and U-NET12 for testing years 2017, 2016, and 2015.

Model	2017	2016	2015	
U-STN12	120 ± 6	120 ± 4	120 ± 3	Prediction Horizon (hrs)
U-NET12	96 ± 8	96 ± 6	96 ± 6	

Referee's comment:

1.4 U-STN is still framed as a physics-informed ML framework
U-STN is still framed as a framework to improve physical consistency:
[Title] Towards physically-consistent [L3 in the abstract] improve their physical consistency
[L44] provide a framework for [L41-42] incorporating physical constraints into the often physics-agnostic ML models
However, given that U-STN does not preserve equivariance (see 1.2), it is unclear why U-STN would make the predictions more physically consistent (other than the qualitative description in Sec 4.1). If it is impossible to quantitatively establish the improved physical consistency of U-STN predictions, I suggest removing the claims listed above.

Authors' response:
Thank you. Based on the referee's suggestion, we have removed these claims. We have changed the word "physical consistency" through-out the manuscript. However, it must be noted that capturing rotation and scaling (albeit in the encoding of the latent space) is motivated from physics. Therefore, we use the word, "physics-inspired". In Lines 41-42 (revised manuscript), we have talked about incorporating physical constraints as a general procedure to improve DDWP models and have not intended to be a description of our DDWP model. We have further clarified in the Lines 75-76 that our DDWP model is not a traditional physics-informed neural network.

Referee's comment:

2 Minor Comments

2.1 Possible discrepancies between the code and the manuscript

In the accompanying code for the baseline U-NETx, it seems that the authors are still training a U-STN ("stn()") for most of the training files instead of a U-NET ("unet_baseline()"). Would it be possible to explain this apparent discrepancy,

which appears in all of the U-NETx (x = 1hr; 12hr, etc.) scripts?

Authors' response:

Thanks for this. We had previously defined the baseline model as stn() as well, just for ease of imports. We have now changed the name of the functions to reflect baseline and U-STN separately. We have accordingly updated both the Github and the Zenodo links.

Referee's comment:

2.2 Typos

[L26-27] "e.g." should be moved to the beginning of the parentheses

[L131] "indicates the Δt that is used": Should e.g., "in units hours" be added to specify the units of Δt in the U-STNx acronym?

[L352] "data-drivenly": Is this grammatically correct or should "in a data-driven fashion" be used instead here?

[L387] "have shown" ->"show"

Authors' response:

Thanks for pointing these out. We have now fixed them based on the referee's suggestions.