We thank all the reviewers for their positive feedback and constructive comments. Below, we first discuss two issues raised by multiple reviewers and then answer and clarify reviewer-specific comments / questions.

Inflation factor α: **R1** and **R4** asked for clarification of the inflation factor α. It is simply the increase in feature dimensions for the `UpDown` model. I.e., if the dimension of q_1 is d then the dimension of q_2 is αd. We will clarify this.

Practical implications: **R3** and **R4** were curious whether our experiments demonstrate practical benefits of our approach. We show state-of-the-art results for rotated MNIST (Fig. 4). However, we have not tackled large-scale learning tasks yet. Our primary contribution is, as acknowledged by **R4**, to *"enrich the understanding and discussion about continuous time deep estimators"* and to offer *"a different viewing angle on the parametrization of neural ODEs."* We are in the process of extending our approach to convolutional networks and to `ResNet`-like structures with striding and feature-dimension changes. This will allow us to explore the performance of our approach for larger-scale tasks.

*Response to **R1***

Paper too dense: This is a fair point. Our aim was to make the main manuscript as self-contained as possible. Hence, we moved many details to the supplementary material. We will, as suggested, add to Fig. 1 an algorithmic description.

Code complexity: We will add a more detailed `README` file and will supplement the existing code with Jupyter notebooks so that a reader can more easily follow. We will also add a simple `PyTorch` example illustrating the approach in parallel with the new algorithmic description and independent of the more complex implementation we submitted.

*Response to **R3***

"Line 5: It's not clear to me why having constant parameters throughout makes a model not continuous-depth" We will rephrase our sentence. The static parameters setting is indeed continuous-depth but obviously does not leverage time varying parameters. **"I don't understand the statement in line 106 with 3d images vs $\mathbb{R}^d$."** In the LDDMM community, the deformation map operates on 2D or 3D spaces, whereas in the DL setting, the map is usually defined in much higher dimensions, e.g. the number of pixels of an image or for us the state-space of our `UpDown` model. **Line 120: Is $R(\Theta(t))$ some corresponding norm?** In our current setting, $R(\theta(t))$ is the Frobenius norm. However, $R(\theta(t))$ can be more general (see, e.g., Appendix A.2 on the Barron norm). We will clarify these points in the manuscript.

*Response to **R4***

"... a convolutional layer formulation is not included" We had, in fact, already implemented shooting for convolutional layers (it is part of the submitted code), but did not include this material and associated experiments to avoid making the paper even more dense.**"It might not be straight forward to apply the shooting formulation to different neural network layers ..."** We agree that there is a trade-off between the ease with which [10] can be deployed for θ constant in time and the increased complexity of our approach for handling dynamic θ. In the "Rotated MNIST" experiment, e.g., our formulation clearly shows benefits over its non-dynamic (`stat. direct` in Fig. 4) counterpart. Hence, increased complexity can improve performance and can therefore be beneficial. Note also that our implementation allows us to automatically derive the shooting equations (Appendix C and implementation) if desired.

"How is this work connected to kernel machines? In particular, as A.2 mentions the connection to reproducing kernel Hilbert spaces." Since our space of vector fields is finite dimensional, it is a proper RKHS when endowed with the Frobenius norm. This connection still needs further exploration using other perspectives present in the DL literature. **"Can a time-dependent θ also be simulated by increasing the state space for static direct, which is closest to the original work of neural ODEs [10]?"** This is an excellent question. Unfortunately, we do not have a theoretically conclusive answer at this point. **"How much more expensive in run-time is the proposed method? Although, only a forward ODE solver run is required, the method needs to track each particle along the way."** Currently, depending on the number of particles, the method is more expensive in terms of run-time compared to [10]. However, our primary focus was on functionality at the current stage, leaving substantial room for optimization and making runtime comparisons difficult at this point. E.g., the matrices in the `UpDown` model are obtained via outer-products. Hence, explicitly forming these matrices could be avoided. Further, for convolutional formulations the particles do not need to have the same dimension as the input images, but could be much smaller patches. For the experiments in the manuscript the `static direct` approach (which does not use any particles) indeed runs faster, but not dramatically so.

"... [the supplementary material] does not include all the experiments ..." We decided to omit any code that would require reviewers to download additional data (e.g., for the "bouncing balls" experiment). Code for *all* experiments will be included in the final version. **"... something wrong with the second part of equation 3.9."** It is actually a renaming of the variables – we overloaded the notations. We will fix this. **"Equations in the appendix seem not to be numbered correctly"** We apologize that equation numbers in the appendix can easily be confused with section numbers. We will fix this in the final version. **"... more discussion [of broader impacts] is needed."** We will add more discussion related to the societal impacts of the algorithms that may result from our work.