Tightness of bounds in Theorem 3.1 (All reviewers) First we note that there is a nontrivial class of networks for which the upper bounds are tight when considering their linear interpolations. Namely, networks with ReLU activation function and mean squared error (MSE) loss. Section C.1 (supp. material) details the proof of Theorem 3.1. The bound depends on the following inequalities; Lipschitz continuity of the activation function, matrix norm inequalities for the layer weights W_i, a triangle inequality when expressing the intermediate activations in layer i as the combination of those in layer $i-1$, a triangle inequality related to the ϵ optimality of the endpoints, and the local Lipschitz continuity of the loss function. For the class of networks mentioned, the last inequality becomes trivial given we have MSE loss. Weights can be found for ReLU networks such that the other inequalities are tight.

We now address the tightness of the bound for piecewise linear curves between networks, which themselves can approximate continuous curves. Under the assumption that a piecewise linear curve or limit of such sequence truly does exist along which the loss is optimal and constant (reasonable as mode connectivity assumes this), for two models on the same line segment of this curve, we can consider the optimal curve between them. It follows that this optimal curve is their linear interpolation. Therefore, we have tightness in the upper bound for curve-finding restricted to piecewise linear curves between a class of networks for which we have tightness in the linear interpolation for that class. Via an argument by continuity, this tightness extends to continuous curves. Thus, these bounds are nontrivial as we have tightness for a wide class of networks and curve parameterizations under a reasonable assumption. We will add the discussion on tightness in the revised version.

Small gap between theory and experiments (R1) We discuss the use of post-activations in section C.2. From Figure 9 (supp. material), it is clear the curve learned with alignment via pre-activations outperforms the unaligned curve. Thus, the theorem applies to a successful method. Since we see best performance from alignment using the correlation of post-activations, we use that method for the majority of the experiments. We discuss the relation between these two techniques in section C.2. To the reviewer's other point, we also note that ReLU is Lipschitz continuous with constant 1.

Neuron Alignment algorithm and symmetry (R1) The symmetry for dense layers, convolutional layers, and residual blocks do have different properties as the reviewer mentions. By conducting experiments on TinyTen, ResNet32, and GoogLeNet, we show that mode connectivity with neuron alignment is able to offer performance gains with these different layers present. We note that Figure 1a, shows that there is still meaningfully symmetry detected by neuron alignment for all of these mentioned layers. Thus, we believe the algorithm can be widely applied.

Sketch of the proof of Theorem 3.1 (R1) We can approximate the intermediate activations of a network along the linear interpolation as an interpolation of the endpoint activations. Then the distance between the activations to those of an endpoint is bounded by the sum of this approximation error and the distance between the two endpoint activations. Thus for aligned networks, this second term in that bound is smaller as the permutation minimizes that value. As we iteratively do this for each layer, we finally bound the distance between the output of the network along the curve to that of the endpoint models, giving us a bound on the error of the network that is tighter with alignment.

Multi-stage alignment (R1) We appreciate the suggestion of a multi-stage neuron alignment. We were curious if models along the curve are aligned to the endpoint models. Figures 1b and 1c show this for the curve midpoint. Interestingly, for aligned curves, the midpoint on the learned curve is already optimally aligned to the endpoints.

Significance of performance (R3) We emphasize our contribution by including the minimal accuracy along the curve in Table 1. We can see that there is a model along the unaligned curves that clearly has subpar performance compared to independently trained models (i.e. the endpoints). Alignment notably improves this. Additionally, we have included figures that show the performance of the linear mode connectivity with and without alignment. This can be seen in Figure 2 (top left) and Figure 4 (supp. material). The performance gain from alignment is very notable, up to 60% difference in minimal accuracy along the curve, in these figures.

Expression of the bounds (R3) The bounds can be stated recursively via the equations in section C.1. We chose to omit it from the main body in the interest of space. B_u is a nonnegative linear combination of the endpoint activation distances, $||f_i^u(0) - f_i^u(1)||_2$. For B_a, the only difference in the expression is the use of $||f_i^u(0) - P_i f_i^u(1)||_2$ for each layer. So when the intermediate activations are closer, as in aligned networks, the upper bound becomes smaller.

Comparisons to Garipov et al. (R4) It seems that there was a misunderstanding. We compare the performance of aligned curves to unaligned curves in our results. When the curves are unaligned, these are precisely curves trained via the method of Garipov et al. Thus our paper shows improvement over existing methods.

Choice of subset for Algorithm 1 (R4) By a subset of the training data, we were referring to a random subset of the training data that well approximates the underlying data manifold. This is to emphasize that the alignment can be computed more quickly on a random subset compared to the whole training dataset. As a test, we computed the alignment for a pair of TinyTen networks trained on CIFAR100, where the subsets contained 2500, 5000, and 10000 images respectively. Despite these being random subsets of differing size, they produced the same alignment.