We thank all reviewers for their excellent feedback. Reviewer 2 noted the absence of an important baseline–using the Kumaraswamy stick-breaking construction with fixed and constant ordering [3]–making it impossible to assess the impact of our work. Our approximate integration over orderings was paramount to semi-supervised learning performance (so much so we overlooked its inclusion). We augmented our results to include some of the requested baselines. Unfortunately, recovering the random number seeds used in our initial experiments was not possible. Therefore, the results for these new baselines (marked with †) will have experienced different initializations and data folds than our original results. However, the test set, from which results are collected, is the same for ALL experiments. Reviewer 2 noted that our tables seemed incomplete. Log likelihood (for unlabeled test data) was not available for our M2 implementation nor the cited models. For clarity, we bifurcated the tables.

Table 1: Classification errors.

Experiment	Method	Error	p-value
MNIST	MV-Kum.	0.103 ± 0.011	$--$
10 trials	IRG†[1]	0.100 ± 0.009	0.51
600 labels	Kumar-SB†[3]	0.247 ± 0.012	1.97×10^{-16}
$\dim(z) = 0$	Softmax	0.100 ± 0.010	0.48
MNIST	MV-Kum.	0.049 ± 0.005	$--$
10 trials	IRG†[1]	0.042 ± 0.004	2.67×10^{-03}
600 labels	M2 (ours)	0.102 ± 0.011	4.17×10^{-11}
$\dim(z) = 2$	Kumar-SB†[3]	0.143 ± 0.015	4.44×10^{-13}
	Softmax	0.053 ± 0.004	0.08
MNIST	MV-Kum.	0.018 ± 0.002	$--$
10 trials	IRG†[1]	0.020 ± 0.004	0.30
600 labels	M2 (ours)	0.020 ± 0.001	0.03
$\dim(z) = 50$	Kumar-SB†[3]	0.077 ± 0.012	8.73×10^{-12}
	Softmax	0.020 ± 0.002	0.01
SVHN	MV-Kum.	0.288 ± 0.025	$--$
4 trials	IRG†[1]	0.291 ± 0.017	0.85
1000 labels	M2 (ours)	0.396 ± 0.010	1.86×10^{-04}
$\dim(z) = 50$	Kumar-SB†[3]	0.707 ± 0.012	8.10×10^{-08}
	Softmax	0.332 ± 0.009	0.02

Table 2: Log likelihoods.

Experiment	Method	Log Likelihood	p-value
MNIST	MV-Kum.	-6.2 ± 5.9	$--$
10 trials	IRG†[1]	-6.1 ± 6.1	0.97
600 labels	Kumar-SB†[3]	-6.4 ± 6.2	0.94
$\dim(z) = 0$	Softmax	-7.2 ± 6.0	0.71
MNIST	MV-Kum.	45.19 ± 0.47	$--$
10 trials	IRG†[1]	45.42 ± 0.45	0.28
600 labels	Kumar-SB†[3]	45.30 ± 0.41	0.56
$\dim(z) = 2$	Softmax	44.19 ± 1.25	0.03
MNIST	MV-Kum.	116.21 ± 0.38	$--$
10 trials	IRG†[1]	116.76 ± 0.39	4.51×10^{-03}
600 labels	Kumar-SB†[3]	116.42 ± 0.40	0.22
$\dim(z) = 50$	Softmax	115.17 ± 0.44	2.36×10^{-05}
SVHN	MV-Kum.	669.69 ± 0.37	$--$
4 trials	IRG†[1]	668.93 ± 0.53	0.06
1000 labels	Kumar-SB†[3]	669.03 ± 0.43	0.06
$\dim(z) = 50$	Softmax	669.55 ± 0.11	0.49

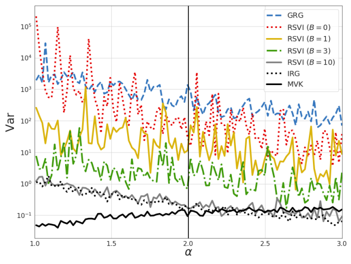

Figure 1: Variance of the ELBO's gradient's first dimension for GRG [4], RSVI [2], IRG [1], and MVK (ours) when fitting a variational posterior to Categorical data with 100 dimensions and a Dirichlet prior. They fit a Dirichlet. We fit a MV-Kumaraswamy.

The drop in classification performance for Kumar-SB [3] is due to the over allocation of probability mass to the final stick during sampling (Fig. 1 of our initial submission)–when the class-assignment posterior has high entropy, the fixed order Monte Carlo integration will bias the last label dimension. We too have included the comparison to Implicit Reparameterization Gradients (IRG) [1]. Here, we set $q(\pi; \alpha_\phi(x)) = \text{Dirichlet}(\pi; \alpha_\phi(x))$ in our semi-supervised model with the same architecture and compute gradients according to [1]. IRG's classification performance is similar to ours and not statistically distinguishable–except for the MNIST $\dim(z) = 2$ experiment, where it likely experienced luckier randomness. Deep learning frameworks' (e.g. TensorFlow, PyTorch, Theano, CNTK, MXNET, Chainer) distinct advantage is NOT requiring user-computed gradients. IRG uses independent Gamma samples to construct Beta and Dirichlet samples. IRG's principle contribution for gradient reparameterization is that it side-steps the need to invert the standardization function (i.e. the CDF). However, IRG still requires Gamma CDF gradients w.r.t. the variational parameters. These gradients do not have a known analytic form and therefore [1] applies forward-mode automatic differentiation to a numerical method. We argue this implementation is not straightforward for the common practitioner. Furthermore, implementing IRG, requires additional (often non-trivial) code to supply IRG gradients to the framework's optimizer. Conversely, our method has analytic gradients, enabling easy integration into ANY deep learning framework. To the best of our knowledge, IRG for the Gamma, Beta, and Dirichlet distributions only exists in TensorFlow (IRG was developed at Deep Mind). It is well established that gradient computations with lower variance are better [1, 2, 4]. We compare our method's gradient variance to these other methods in fig. 1. Reviewer 2, we did not test Gamma-SB [3] as they demonstrate inferiority to Kumar-SB (in the context of their non-parametric model's reconstruction error). Reviewer 1 is correct regarding π and x in lines 173 & 181. Reviewer 1, We agree our KL-Divergence discussion is not clear. Let $\alpha_\phi^{(j)}(x)$ be the j^{th} concentration parameter of the inference network, and $\alpha^{(j)}$ be j^{th} parameter of the Dirichlet prior. Then, for the set of all orderings O, $D_{KL}(q(\pi; \alpha_\phi(x)) \,||\, p(\pi; \alpha)) =$

$$D_{KL}(\text{MV-Kumaraswamy}(\alpha_\phi(x)) \,||\, \text{Dirichlet}(\alpha)) = \mathop{\mathbb{E}}_{o \sim \text{Uniform}(O)} \left[\sum_{i=1}^{K-1} D_{KL}\Big(\text{Kumaraswamy}\left(\alpha_\phi^{(o_i)}(x), \sum_{j=i+1}^{K} \alpha_\phi^{(o_j)}(x)\right) \Big|\Big| \text{Beta}\left(\alpha^{(o_i)}, \sum_{j=i+1}^{K} \alpha^{(o_j)}\right)\Big)\right]$$

We compute $D_{KL}(\text{Kumaraswamy}(a, b) \,||\, \text{Beta}(a', b'))$ analytically as in [3] with a 5th order Taylor approx. We suspect approximate symmetry arises with $O(K^2)$ samples vs. $K!$ for full symmetry. Since the problematic bias occurs on the last stick, each label needs an ordering where it is not last, which has probability $K = \frac{K!}{(K-1)!}$ of occurring. Thus, for this to occur for all labels, we have K^2. Reviewer 2, we set $K = 10$, the number of classes–not 50 as you suggest. Reviewer 2 notes we could replace the Dirichlet prior with a MV-Kumaraswamy prior–we agree, however, chose the Dirichlet prior for reader familiarity and because we are proposing a Dirichlet surrogate. Reviewer 2, if allowed, we would be happy to change the title to "A New, Explicitly Reparameterizable Surrogate for the Dirichlet."

References

[1] Mikhail Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit reparameterization gradients. In *Advances in Neural Information Processing Systems 31*, pages 441–452. 2018.

[2] Christian A. Naesseth, Francisco J. R. Ruiz, Scott W. Linderman, and David M. Blei. Reparameterization gradients through acceptance-rejection sampling algorithms. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

[3] Eric Nalisnick and Padhraic Smyth. Stick-breaking variational autoencoders. *International Conference on Learning Representations (ICLR)*, Apr 2017.

[4] Francisco R Ruiz, Michalis Titsias RC AUEB, and David Blei. The generalized reparameterization gradient. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 460–468. Curran Associates, Inc., 2016.