Appendix for
Variational Auto-Regressive Gaussian Processes for Continual Learning

Sanyam Kapoor [1] **Theofanis Karaletsos** [2] **Thang D. Bui** [3]

A. VAR-GPs

A.1. Posterior Predictive

For a novel input $\mathbf{x}_\star$, the posterior predictive is computed via a Monte Carlo approximation of

$$p(y_\star \mid \mathbf{x}_\star) = \int p(y_\star \mid f) q_t(f, \theta \mid \mathbf{x}_\star) df d\theta \tag{12}$$

For a K-way classifier, we train K independent GPs and use the Bayes optimal prediction $\arg\max_i p(y_\star^{(i)} \mid \mathbf{x}_\star)$, for all $i \in \{1, \ldots, K\}$, to compute accuracies.

B. Ablations

B.1. Global Inducing Points

This section outlines the assumptions made for the ablation titled as "Global". The characterization for the first task remains the same as in VAR-GPs. For subsequent tasks, the general model and the variational assumption is written as (with implicit dependence on $\mathbf{Z}$),

$$\begin{aligned} p(\mathbf{y}^{(t)}, f, \theta \mid \mathbf{X}^{(t)}, \mathcal{D}^{(<t)}) \approx \\ \prod_{i=1}^{N_t} p(y_i^{(t)} \mid f, \mathbf{x}_i^{(t)}) \\ p(f_{\neq \mathbf{u}_{t-1}} \mid \mathbf{X}^{(t)}, \mathbf{u}_{t-1}, \theta) \\ q(\mathbf{u}_{t-1}) q_{t-1}(\theta) \, . \end{aligned} \tag{13}$$

Note that we don't have the auto-regressive characterization of VAR-GPs in the model anymore and instead have an approximate dependence through the variational posterior for the previous task. We further note that,

$$\begin{aligned} p(f_{\neq \mathbf{u}_{t-1}} \big| \mathbf{X}^{(t)}, \mathbf{u}_{t-1}, \theta) = p(f_{\neq \mathbf{u}_{t-1}, \mathbf{u}_t} \big| \mathbf{X}^{(t)}, \mathbf{u}_{t-1}, \mathbf{u}_t, \theta) \\ \frac{p(\mathbf{u}_{t-1}, \mathbf{u}_t \big| \theta)}{p(\mathbf{u}_{t-1} \big| \theta)}, \end{aligned} \tag{14}$$

$$\begin{aligned} p(f_{\neq \mathbf{u}_t} \big| \mathbf{X}^{(t)}, \mathbf{u}_t, \theta) = p(f_{\neq \mathbf{u}_{t-1}, \mathbf{u}_t} \big| \mathbf{X}^{(t)}, \mathbf{u}_{t-1}, \mathbf{u}_t, \theta) \\ \frac{p(\mathbf{u}_{t-1}, \mathbf{u}_t \big| \theta)}{p(\mathbf{u}_t \big| \theta)}. \end{aligned} \tag{15}$$

Owing to key cancellations, the variational lower bound now is given by,

$$\begin{aligned} \mathcal{F}(q_t) = \sum_{i=1}^{N_t} \mathbb{E}_{q_t(f,\theta)} \left[\log p \left(y_i^{(t)} \mid f, \mathbf{x}_i^{(t)} \right) \right] \\ - \mathcal{KL} \left[q_t(\theta) \mid\mid q_{t-1}(\theta) \right] \\ - \mathbb{E}_{q_t(\theta)} \left[\mathcal{KL} \left[q(\mathbf{u}_t) \mid\mid p(\mathbf{u}_t \mid \theta) \right] \right] \\ + \mathbb{E}_{q_t(\theta) q(\mathbf{u}_t \mid \theta) p(\mathbf{u}_{t-1} \mid \mathbf{u}_t, \theta)} \left[\log \frac{q(\mathbf{u}_{t-1})}{p(\mathbf{u}_{t-1} \mid \theta)} \right] . \end{aligned} \tag{16}$$

A key difference to note here is that the $\mathcal{KL}$ regularizer containing inducing points $\mathbf{u}_t$ is not conditional on the previous ones anymore.

B.2. Re-VAR-GP: Retraining old inducing points

In this version of VAR-GP, we allow retraining of old inducing points and call it Retrainable VAR-GP (abbreviated as Re-VAR-GP). We clarify the precise nature of terms. Leading from (6) and (8), we highlight frozen $\widetilde{\mathbf{u}}_{<t}$ and $\widetilde{\mathbf{Z}}_{<t}$ (with a tilde) in the prior model to differentiate against learnable parameters,

$$\begin{aligned} p(\mathbf{y}^{(t)}, f, \theta \mid \mathbf{X}^{(t)}, \mathcal{D}^{(<t)}) = \prod_{i=1}^{N_t} p(y_i^{(t)} \mid f, \mathbf{x}_i^{(t)}) \\ p(f_{\neq \mathbf{u}_t, \widetilde{\mathbf{u}}_{<t}} \mid \mathbf{X}^{(t)}, \mathbf{u}_t, \widetilde{\mathbf{u}}_{<t}, \mathbf{Z}_t, \widetilde{\mathbf{Z}}_{<t}, \theta) \\ p(\mathbf{u}_t \mid \mathbf{Z}_t, \widetilde{\mathbf{u}}_{<t}, \widetilde{\mathbf{Z}}_{<t}, \theta) \\ q(\widetilde{\mathbf{u}}_{<t} \mid \widetilde{\mathbf{Z}}_{<t}, \theta) q_{t-1}(\theta) \, . \end{aligned} \tag{17}$$

We posit the variational posterior as,

$$\begin{aligned} q_t(f, \theta) = p(f_{\neq \mathbf{u}_t, \mathbf{u}_{<t}} \mid \mathbf{X}^{(t)}, \mathbf{u}_t, \mathbf{u}_{<t}, \mathbf{Z}_t, \mathbf{Z}_{<t}, \theta) \\ q(\mathbf{u}_t \mid \mathbf{Z}_t, \mathbf{u}_{<t}, \mathbf{Z}_{<t}, \theta) \\ q(\mathbf{u}_{<t} \mid \mathbf{Z}_{<t}, \theta) q_t(\theta) \end{aligned} \tag{18}$$

To simplify these equations, we note the following identi-

ties.

$$
\begin{aligned}
&p(f_{\neq \mathbf{u}_t, \mathbf{u}_{<t}} \mid \mathbf{X}^{(t)}, \mathbf{u}_t, \mathbf{u}_{<t}, \mathbf{Z}_t, \mathbf{Z}_{<t}, \theta) = \\
&p(f_{\neq \mathbf{u}_t, \mathbf{u}_{<t}, \tilde{\mathbf{u}}_{<t}} \mid \mathbf{X}^{(t)}, \mathbf{u}_t, \mathbf{u}_{<t}, \tilde{\mathbf{u}}_{<t}, \mathbf{Z}_t, \mathbf{Z}_{<t}, \tilde{\mathbf{Z}}_{<t}, \theta) \\
&p(\tilde{\mathbf{u}}_{<t} \mid \mathbf{u}_t, \mathbf{u}_{<t}, \mathbf{Z}_t, \mathbf{Z}_{<t}, \tilde{\mathbf{Z}}_{<t}, \theta) \,,
\end{aligned}
$$

$$
\begin{aligned}
&p(f_{\neq \mathbf{u}_t, \tilde{\mathbf{u}}_{<t}} \mid \mathbf{X}^{(t)}, \mathbf{u}_t, \tilde{\mathbf{u}}_{<t}, \mathbf{Z}_t, \tilde{\mathbf{Z}}_{<t}, \theta) = \\
&p(f_{\neq \mathbf{u}_t, \mathbf{u}_{<t}, \tilde{\mathbf{u}}_{<t}} \mid \mathbf{X}^{(t)}, \mathbf{u}_t, \mathbf{u}_{<t}, \tilde{\mathbf{u}}_{<t}, \mathbf{Z}_t, \mathbf{Z}_{<t}, \tilde{\mathbf{Z}}_{<t}, \theta) \\
&p(\mathbf{u}_{<t} \mid \mathbf{u}_t, \tilde{\mathbf{u}}_{<t}, \mathbf{Z}_t, \mathbf{Z}_{<t}, \tilde{\mathbf{Z}}_{<t}, \theta) \,,
\end{aligned}
$$

$$
\begin{aligned}
&\frac{p(\mathbf{u}_t \mid \mathbf{Z}_t, \tilde{\mathbf{u}}_{<t}, \tilde{\mathbf{Z}}_{<t}, \theta) p(\mathbf{u}_{<t} \mid \mathbf{u}_t, \tilde{\mathbf{u}}_{<t}, \mathbf{Z}_t, \mathbf{Z}_{<t}, \tilde{\mathbf{Z}}_{<t}, \theta)}{p(\tilde{\mathbf{u}}_{<t} \mid \mathbf{u}_t, \mathbf{u}_{<t}, \mathbf{Z}_t, \mathbf{Z}_{<t}, \tilde{\mathbf{Z}}_{<t}, \theta)} \\
&= \frac{p(\mathbf{u}_{<t}, \mathbf{u}_t \mid \mathbf{Z}_t, \mathbf{Z}_{<t}, \theta)}{p(\tilde{\mathbf{u}}_{<t} \mid \tilde{\mathbf{Z}}_{<t}, \theta)}.
\end{aligned}
$$

Using these identities, the lower bound now simplifies as,

$$
\begin{aligned}
\mathcal{F}(q_t) = &\sum_{i=1}^{N_t} \mathbb{E}_{q_t(f,\theta)} \left[\log p(y_i^{(t)} \mid f, \mathbf{x}_i^{(t)}) \right] \\
&- \mathcal{KL}\left[q_t(\theta) \,||\, q_{t-1}(\theta) \right] \\
&- \mathbb{E}_{q_t(\theta)} \left[\mathcal{KL}\left[q(\mathbf{u}_{\leq t} \mid \mathbf{Z}_{\leq t}, \theta) \,||\, p(\mathbf{u}_{\leq t} \mid \mathbf{Z}_{\leq t}, \theta) \right] \right] \\
&- \mathbb{E}_{q_t(\theta) q(\mathbf{u}_{\leq t} \mid \mathbf{Z}_{\leq t}, \theta) p(\tilde{\mathbf{u}}_{<t} \mid \tilde{\mathbf{Z}}_{<t}, \mathbf{u}_{\leq t}, \mathbf{Z}_{\leq t}, \theta)} \left[\mathfrak{R}_t \right] \,,
\end{aligned}
\tag{19}
$$

where $\mathfrak{R}_t = \log \frac{p(\tilde{\mathbf{u}}_{<t} \mid \tilde{\mathbf{Z}}_{<t}, \theta)}{q(\tilde{\mathbf{u}}_{<t} \mid \tilde{\mathbf{Z}}_{<t}, \theta)}$. The key differences to note here are the fact that prior model now conditions on the frozen inducing points while the new variational distributions introduced are still free to optimize those points further. This leads to additional terms in the variational lower bound. We briefly discuss the performance next.

B.2.1. Performance on Toy Dataset

Similar in spirit to Figure 1, we train Re-VAR-GP on the toy dataset in Figure 2. The density plots for training after both first task (training on classes $0/1$) and the second (training on classes $2/3$) are presented in Figure 7.

As shown in Figure 7, Re-VAR-GP is not able to retain the information gained from previous task, a sign of catastrophic forgetting. This can be understood from the nature of the lower bound in (19). The only term that can potentially contribute to preservation of old information is the expected ratio $\log \frac{p(\tilde{\mathbf{u}}_{<t} \mid \tilde{\mathbf{Z}}_{<t}, \theta)}{q(\tilde{\mathbf{u}}_{<t} \mid \tilde{\mathbf{Z}}_{<t}, \theta)}$. However, this term avoids any interaction between $\tilde{\mathbf{u}}_{\leq t}$ and $\mathbf{u}_{\leq t}$. As a result, the retrainable parameters $\mathbf{u}_{\leq t}$ and $\mathbf{Z}_{\leq t}$ have no information-preserving regularization unlike VAR-GPs as seen in (8). Owing to this observation, we do not pursue this model further.

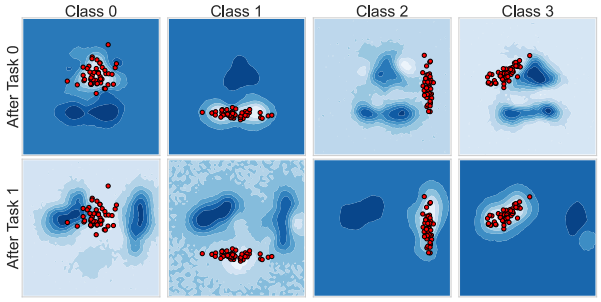

Figure 7. This figure shows class-wise output probabilities in each column for classifiers trained using a synthetic dataset (Figure 2) on the 2-D plane $x, y \in [-3., 3.]$. The first row represents the density surface after training for **Task 0** (observing classes $0/1$) and the second after training for **Task 1** (observing classes $2/3$). Brighter regions represent higher probabilities. Training points for each class are marked ●. Re-VAR-GP tends to suffer from catastrophic forgetting. Notice the approximately uniform uncertainty in regions for Class 0 and Class 1 after training on the second task.

B.3. Deep Kernel Learning

For increased representational power in the kernel, we also provide preliminary experiments with Deep Kernel Learning (Wilson et al., 2016). Effectively, we augment the Exponentiated Quadratic kernel with a feature extractor $g_\phi(x)$ in the form of a neural network and allow them to be trained with additional kernel hyperparameters ϕ. This amounts to replacing $\mathbf{x}$ and $\mathbf{x}'$ in (20) with $g_\phi(\mathbf{x})$ and $g_\phi(\mathbf{x}')$ respectively. We only use point estimates for ϕ, initialized at the previous task for all $t > 1$.

B.3.1. Experiments with Split MNIST

We use a neural network with two hidden layers of size 256 each and the final output feature size of 64 to parameterize f_ϕ and train the system end-to-end. As we see in Figure 8, the performance declines much faster than in VAR-GPs.

This result hints towards weak regularization of the feature extractor as we encounter subsequent tasks. The introduction of a neural network makes the inference problem much harder and any potential remedies are beyond the scope of current work. We, therefore, do not discuss this further but makes for an exciting direction to pursue in the future.

C. Implementation

C.1. Exponentiated Quadratic kernel

The precise parametrization of the kernel used is given below. $\|\cdot\|_2$ is the ℓ^2-norm. We parameterize $\log \gamma$ and $\log \sigma$.

$$
k(\mathbf{x}, \mathbf{x}') = \gamma \exp\left\{ \frac{||\mathbf{x} - \mathbf{x}'||_2^2}{2\sigma^2} \right\} \tag{20}
$$

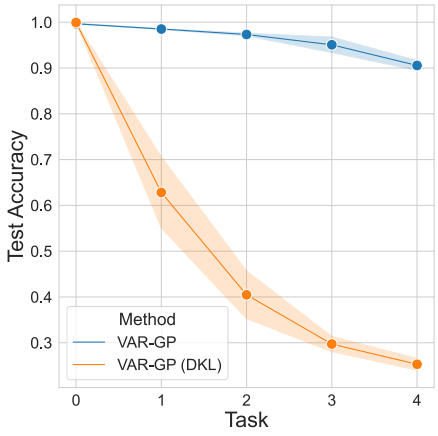

Figure 8. This figure shows the task-wise test accuracy on Split MNIST over five independent runs. We train a neural network as a feature extractor before applying the kernel (as described in Appendix B.3). It is clear that the neural networks require stronger regularization as we incorporate more tasks.

C.2. Parameterizing covariance matrices

In all experiments, we parameterize a covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{M \times M}$ using its Cholesky decomposition $\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^\intercal$, where $\mathbf{L} \in \mathbb{R}^{M \times M}$ is a lower triangular matrix with positive diagonals. The positivity of the diagonals is maintained via a `softplus` transform. As a result, we can apply unconstrained optimization on $\frac{1}{2}(M \times (M+1))$ free parameters corresponding to the lower trianglular matrix $\mathbf{L}$.

C.3. Computing the auto-regressive distributions in VAR-GPs

When using the auto-regressive parametrization in VAR-GPs, the joint distribution over all inducing points up to and including the current time step can be decomposed as follows,

$$\begin{aligned} q(\mathbf{u}_{\leq t}|\theta) &= q(\mathbf{u}_{<t}|\theta) q(\mathbf{u}_t|\mathbf{u}_{<t}, \theta) \\ &= \mathcal{N}(\mathbf{u}_{<t}; \mathbf{m}_{<t}, \mathbf{\Sigma}_{<t}) \\ &\quad \mathcal{N}(\mathbf{u}_t; \mathbf{A}_t \mathbf{u}_{<t} + \mathbf{m}_t, \mathbf{\Sigma}_t) \,, \end{aligned} \tag{21}$$

such that $\mathbf{A}_t = \mathbf{K}_{\mathbf{Z}_t, \mathbf{Z}_{<t}} \mathbf{K}^{-1}_{\mathbf{Z}_{<t}, \mathbf{Z}_{<t}}$.

While we cannot avoid sampling the hyperparameters θ, we can avoid variance introduced by the ancestral sampling of variational distribution for computation of (8). We recognize that the full auto-regressive distribution can be computed in closed form as it is a product Gaussians with linear dependence in the mean, similar in spirit to linear Gaussian dynamical systems (Murphy, 2012). Hence, for all $t > 1$, we have

$$q(\mathbf{u}_{\leq t}|\theta) = \mathcal{N}\left(\begin{bmatrix} \mathbf{u}_{<t} \\ \mathbf{u}_t \end{bmatrix}; \begin{bmatrix} \mathbf{m}_{<t} \\ \mathbf{A}_t \mathbf{m}_{<t} + \mathbf{m}_t \end{bmatrix}, \mathbf{\Sigma} \right) \tag{22}$$

where,

$$\mathbf{\Sigma} \triangleq \begin{bmatrix} \mathbf{\Sigma}_{<t} & \mathbf{\Sigma}_{<t} \mathbf{A}_t^\intercal \\ \mathbf{A}_t \mathbf{\Sigma}_{<t}^\intercal & \mathbf{\Sigma}_t + \mathbf{A}_t \mathbf{\Sigma}_{<t} \mathbf{A}_t^\intercal \end{bmatrix}$$

D. Hyperparameters

D.1. Search Space

The search space for all hyperparameters used across experiments is described in Table 2. Top hyperparameters were picked using a held-out validation set.

Table 2. List of key hyperparameters with relevant search spaces.

Hyperparameter	Range / Value
Learning Rate (η)	$[0.001, 0.01]$
Inducing Points (M)	$[40, 200]$
Hypers $\mathcal{KL}$ Tempering Factor (β)	$[1.0, 10.0]$
Batch Size (B)	512
Maximum Epochs (E)	500
Early Stopping Patience Epochs (K)	200
Early Stopping Tolerance (δ)	0.0001

D.2. Varying number of inducing points M

In Figure 9, we note the mean performance by varying the number of inducing points M from 20 to 200 in steps of 20, for Split MNIST. The key takeaway here is that increasing the number of inducing points generally does improve the performance. This indicates that there may be more capacity available to be exploited further.

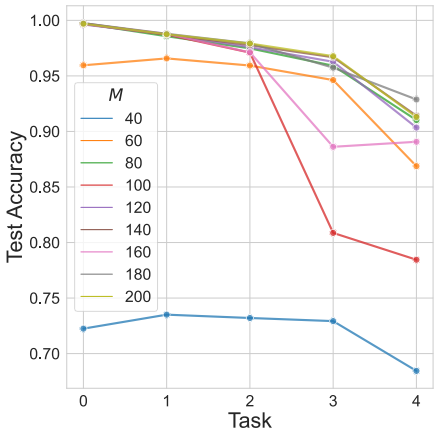

Figure 9. This figure shows the mean task-wise (on x-axis) test accuracy (y-axis) on Split MNIST over five independent runs, varying number of inducing points M from 20 to 200 in steps of 20. The key insight to draw here is about the general trend that increasing the number of inducing points M improves the mean peformance, hinting towards more capacity being available to be exploited by the learning algorithm.