# Fleet Control using Coregionalized Gaussian Process Policy Iteration – Supplementary Information

**Timothy Verstraeten**[1] and **Pieter J.K. Libin**[1] and **Ann Nowé**[1]

## Appendix A – Covariance Matrix Inversion

The inversion of a covariance matrix is necessary when computing the posterior statistics of the GPs, i.e.,

$$\begin{aligned}\mathbb{E}\left[\mathbf{f} \mid X, X_{\text{tr}}, \mathbf{y}_{\text{tr}}\right] &= K_{X,X_{\text{tr}}} C^{-1}_{X_{\text{tr}},X_{\text{tr}}} \mathbf{y}_{\text{tr}} \\ \mathbb{V}\left[\mathbf{f} \mid X, X_{\text{tr}}, \mathbf{y}_{\text{tr}}\right] &= K_{X,X} - K_{X,X_{\text{tr}}} C^{-1}_{X_{\text{tr}},X_{\text{tr}}} K_{X_{\text{tr}},X} \\ C_{X_{\text{tr}},X_{\text{tr}}} &= K_{X_{\text{tr}},X_{\text{tr}}} + \sigma^2 I,\end{aligned} \tag{1}$$

where $K_{X,X_{\text{tr}}}$ is a matrix containing the pair-wise covariances between sets $X$ and $X_{\text{tr}}$ according to the covariance kernel and $\sigma^2$ is observational noise. In the literature, inversion is achieved by performing backward and forward substitution on the Cholesky factors of the covariance matrix, as it faster and more numerically stable than direct inversion [2]. This operation has a complexity of $O(N^3)$, where $N$ signifies the number of rows. In this section, we provide a faster operation for our sparse coregionalization model.

Consider a fleet of $M$ members. Each member $m$ has a data set of $N_m$ samples, resulting in a total of $N = \sum_{m=1}^{M} N_m$ samples. Without loss of generality, assume that the target index is $M$ and the sources are in $[1, \ldots, (M-1)]$. When we use the transition model described in Section 5 in the main paper, a covariance matrix needs to be inverted when fitting the fleet-wide GP. This covariance matrix has the following block form:

$$K^F = \begin{bmatrix} B & C \\ C^T & K^{\text{SE}}_M \end{bmatrix}, \tag{2}$$

with block diagonal matrix

$$B = \begin{bmatrix} K^{\text{SE}}_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & K^{\text{SE}}_{M-1} \end{bmatrix}, \tag{3}$$

where $K^{\text{SE}}_m$ is the covariance matrix of member $m$ and $C$ contains the cross-covariance matrices between the target and each of the sources.

The Cholesky decomposition [2] of the full covariance matrix $K^F$ is

$$K^F = L_{K^F} L^T_{K^F} = \begin{bmatrix} L_B & 0 \\ (L^{-1}_B C)^T & L_S \end{bmatrix} \begin{bmatrix} L^T_B & L^{-1}_B C \\ 0 & L^T_S \end{bmatrix} \tag{4}$$

where $L_B$ and $L_S$ are the Cholesky factors of block $B$ and its Schur complement:

$$\begin{aligned} S &= K^{SE}_M - C^T B^{-1} C \\ &= K^{SE}_M - C^T (L_B L^T_B)^{-1} C \\ &= K^{SE}_M - (L^{-1}_B C)^T L^{-1}_B C. \end{aligned} \tag{5}$$

Note that

$$L_B = \begin{bmatrix} L_{K^{\text{SE}}_1} & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & L_{K^{\text{SE}}_{M-1}} \end{bmatrix}. \tag{6}$$

The only matrices for which a Cholesky factor needs to be computed are matrices $S$ and $K^{\text{SE}}_m$. The Cholesky decomposition has a cubic complexity in the number of rows. Thus, as each block in B can be processed separately, the computational complexity of the Cholesky decomposition on matrix B is $O\left(\sum_{m=1}^{M} N^3_m\right)$. The size of $S$ is $N_M \times N_M$, and thus, computing its Cholesky factor is equal to $O\left(N^3_M\right)$. As the multiplication of $L_B C$ can also be decomposed per member (i.e., per block in the matrix $L_B$), the complexity of this operation is $O\left(\sum_{m=1}^{M} N_M N^2_m\right)$. Thus, the combined complexity of our sparse matrix inversion is $O\left(\sum_{m=1}^{M} N^3_m\right)$, which is significantly smaller than the cubic complexity on the full data set, i.e., $O\left(\left(\sum_{m=1}^{M} N_m\right)^3\right)$.

## Appendix B – Sensitivity Analysis

We perform a sensitivity analysis on the fleet variant of the continuous mountain car domain, proposed in the main paper, to investigate the robustness of our fleet-wide policy iteration method (see Section 7 in the main paper). In this problem, an action is a force applied to either side of the car. The magnitude of this force (i.e., the power) can be configured. We consider a fleet of three mountain cars: a target, source A and source B. We use the same parameters for the setting as described in the main paper, except that we vary the power parameter of source A. Specifically, the target has a power of $15 \cdot 10^{-4}$, source B has a power of $10^{-4}$ units, and source A has a power that varies according to $(5+i) \cdot 10^{-4}$, for $i \in [0..10]$. This ranges captures various grades of similarity between the target and source A.

We run the experiment 50 times for the three target types: single, joint and fleet. We measure performance in terms of the sum of squared distances to the goal. We repeat each experiment 50 times. The results are shown in Figure 1. As expected, the fleet target performs better when source A has a power closer to the target's power. When the power of source A decreases, there is less relevant information for source A to share with the target. This results in similar

[1] Artificial Intelligence Lab Brussels,
Vrije Universiteit Brussel, Belgium,
email: {tiverstr, plibin, anowe}@vub.be

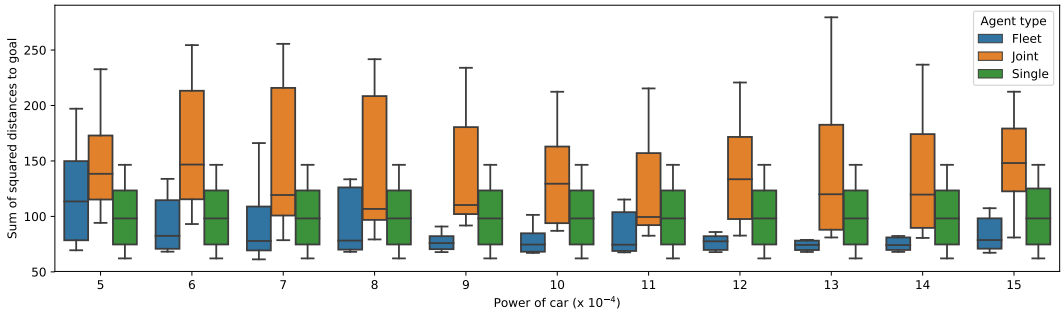



**Figure 1**: Sensitivity analysis – Boxplot of the sum of squared distances to the goal over 200 time steps for 50 runs.

outcomes as for the single target type, which does not perform any transfer. Still, we can see that the fleet target significantly outperforms both the joint and single targets for the higher power levels, and exhibits a performance similar to the single target for the lower power levels. The discrepancy between the performance of the single and fleet targets at a power level of $5 \cdot 10^{-4}$ is expected, as the single target already assumes there is no correlation between the target and the sources, while our fleet target still needs to learn this fact. As additional accurate domain knowledge is available to the single target, the problems becomes strictly easier for the single target to solve. Naturally, such domain knowledge will not be available in the real world, highlighting the need for a flexible method as our own.

## Appendix C – Success Rates

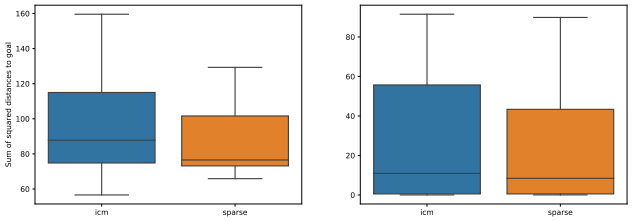
In Table 1, we show the percentages of success for both the mountain car and cart-pole benchmarks over 50 runs. For mountain car, a successful run is one where the car reaches the goal within 200 time steps, while for cart-pole, a successful run is one where the cart manages to keep the pole above the threshold for 200 time steps. Although the percentage of success for both the single and fleet targets are high, we show in the main manuscript that the performance (measured by the sum of the squared distances from the goal) of a policy learned by the single target type is much higher than the performance of a policy learned by the fleet target type. This means that the single target type infers a sub-optimal policy for reaching the goal, while the fleet target type often exhibits optimal behavior.

| | joint | single | fleet |
|---|---|---|---|
| mountain car | 64% | 80% | 86% |
| cart-pole | 38% | 74% | 86% |

**Table 1**: Success rates

## Appendix D – Intrinsic Coregionalization Model

The transition model described in the main manuscript is a sparse variant of the intrinsic coregionalization model [1]. As discussed in Appendix A, the computational complexity is significantly decreased when using our sparse model, compared to a fully-connected model such as the intrinsic coregionalization model. To show that using our sparse transition model does not incur a significant decrease in performance, we compare our model (sparse) against the intrinsic coregionalization model (icm) on the same synthetic benchmarks as in the main manuscript (see Figure 2). Note that the results for the sparse model are the same as for the fleet target type reported in the main manuscript.




(a) Mountain car (b) Cart-pole

**Figure 2**: Boxplot of the total sum of squared distances to the goal state for the mountain car (a) and cart-pole (b) benchmarks during 200 time steps for 50 runs.

## REFERENCES

[1] Edwin V. Bonilla, Kian Ming A. Chai, and Christopher K. I. Williams, 'Multi-Task Gaussian Process Prediction', *Advances in Neural Information Processing Systems*, **20**, 153–160, (2008).

[2] Carl Edward Rasmussen and Christopher K. I. Williams, *Gaussian Processes for Machine Learning*, The MIT Press, Cambridge, MA, USA, 2006.