...In this task, you should use Bayesian optimization to tune one hyperparameter of a machine learning model subject to a constraint on a property of the model. Let $\theta \in \Theta$ denote the hyperparameter of interest, e.g., the number of layers in a deep neural network. We aim at finding a network that makes accurate and fast predictions. To this end, we can train our model for a specific value of θ on a training data set. From this, we obtain a corresponding accuracy on the validation data set and an average prediction speed. In this context, our goal is to find θ^*, i.e., the value of the hyperparameter that induces the highest possible validation accuracy, such that a requirement on the average prediction speed is satisfied.

More formally, let us denote with $f : \Theta \to [0, 1]$ the mapping from the space of hyperparameter to the corresponding validation accuracy. When we train with a given θ, we observe $y_f = f(\theta) + \varepsilon_f$, where $\varepsilon_f \sim \mathcal{N}(0, \sigma_f^2)$ is zero mean Gaussian i.i.d. noise. Moreover, we denote with $v : \Theta \to \mathbb{R}^+$ the mapping from the space of hyperparameters to the corresponding prediction speed. Similar to the accuracy, we observe a noisy value of the speed, which we denote with $y_v = v(\theta) + \varepsilon_v$, with $\varepsilon_v \sim \mathcal{N}(0, \sigma_v^2)$ zero mean Gaussian i.i.d. noise. Then, the problem is formalized as,

$$\theta^* \in \mathrm{argmax}_{\theta \in \Theta, v(\theta) > \kappa} f(\theta),$$

where κ is the minimum tolerated prediction speed. The objective of this problem does not admit an analytical expression, is computationally expensive to evaluate and is only accessible through noisy evaluations. Therefore, it is well suited to Bayesian optimization (see [1] for further reading on Bayesian optimization for hyperparameter tuning).

In this project, you need to solve the hyperparameter tuning problem presented above with Bayesian optimization. Let θ_i be the hyperparameter evaluated at the i^{th} iteration of the Bayesian optimization algorithm. **While running the Bayesian Optimization algorithm, you may try hyperparameters for which the speed constraint is violated, i.e. $v(\theta_i) < \kappa$. However, the final solution must satisfy $v(\theta) \geq \kappa$.**

Remarks: In the motivating example above, θ takes discrete values (number of layers is a Natural number) and the objective and the constraint can be evaluated independently. However, to keep the problem simple, we let θ be continuous and we evaluate f and v simultaneously. Moreover, to avoid unfair advantages due to differences in computational power, the training of the neural network is simulated and, therefore, the time required for this step is platform independent. This task does not have a private score.

Below, you can find the quantitative details of this problem.

- The domain is $\Theta = [0, 5]$.
- The noise perturbing the observations is Gaussian with standard deviation $\sigma_f = 0.15$ and $\sigma_v = 0.0001$ for the accuracy and the speed, respectively.
- The mapping f can be effectively modeled with a Matérn (https://en.wikipedia.org/wiki/Mat%C3%A9rn_covariance_function) kernel with variance 0.5, lengthscale 0.5 and smoothness parameter $\nu = 2.5$.
- The mapping v can be effectively modeled with a constant mean of 1.5 and a Matérn (https://en.wikipedia.org/wiki/Mat%C3%A9rn_covariance_function) kernel with variance $\sqrt{2}$, lengthscale 0.5, and smoothness parameter $\nu = 2.5$.
- The minimum tolerated speed is $\kappa = v^{\min} = 1.2$. The unit of measurement is not relevant as the training is only simulated.