Guassian Processes/Robot Arm Dynamics

Jacob Lister

Department of Electrical and Computer Engineering
Department of Mathematical Sciences
Purdue Fort Wayne

Introduction

Inverse dynamics is used when working to model joint movement, often of a robotic arm or the arms and legs of an animal or person. This technique is used to determine the torque being applied to the joints in a system based on the position, velocity, and accelation of a body (or end effector in the case of a robotic arm). Using inverse dynamics, you can compute the torques needed to follow a certain trajectory, which provides the ability to simulate biological models or allows robotic arms to "plan" the path they will take when moving from one position to another.

The specific system I am looking at is that of a SARCOS robotic arm. This arm has 7 joints, which means 7 torques to find. Additionally, this means that there are 7 positions, velocities, and accelerations as outputs. Because I was working with inverse dynamics, the positions, velocities, and accelerations will be used as inputs, and the torques will be used as outputs. This means I will be creating a mapping of 21 inputs of 7 outputs.

Inverse dynamics can be performed in a variety of ways. The most straight-forward way is to analytically compute the torque functions of each joint in the system. This method requires full knowledge of the system and often becomes extremely complicated for many-jointed systems, so this method cannot always be used. There are also methods of estimating the torques using statistical methods. Due to the fact that the composition of torque functions is of trigonometric functions mainly, these functions end up being highly non-linear. Due to the non-linearity of the torques, simple statistic methods such as linear regression are expected to perform poorly. Guassian Processes Regression is a more robust statistical method and will be the focus of this work. Guassian Processes Regression was used to model the torques of the SARCOS arm given the 21 inputs.

GPR

To estimate the torque functions of the SARCOS arm, I used a Guassian Processes Regression (GPR). GPR is a technique used in machine learning and more broadly in statistics. It is often used a supervised learning technique when working with nonlinear systems. Because Guassian Processes (GPs) can interpreted as a distribution over a function space we can use them to create a more robust method of regression. Because of the fact that the dataset I am working with is large, having over 40,000 samples, an approximation method for GPR will be used.

Now for an in-depth explanation of GPR. If we interpret a GP as being a distribution over a function space, we can write it as:

$$\begin{aligned} f(\mathbf{x}) &\sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \\ \mathbb{E}[f(\mathbf{x})] &= m(\mathbf{x}) \\ \text{cov}[f(\mathbf{x}), f(\mathbf{x}')] &= k(\mathbf{x}, \mathbf{x}') \end{aligned}$$

where $\mathbf{x}$, $\mathbf{x}'$ are 2 points chosen from the set of possible inputs. Then, if we have a number of input points X_*, we can generate a random Guassian vector $\mathbf{f}_* = f(\mathbf{x}_*) \sim \mathcal{N}(0, K(X_*, X_*))$, where K is the covariances matrix between the different inputs. This becomes the prior for the function we are trying to approximate. If we take into account noise and condition the prior on the observations (training data), we have:

$$\begin{aligned} &\mathbf{f}_* \mid X_*, X, \mathbf{f} \sim \mathcal{N}(K(X_*, X)K(X, X)^{-1}\mathbf{f}, \\ &\quad K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*)) \end{aligned}$$

This means that by evaluating the mean and covariance matrices above, the function values $\mathbf{f}_*$ corresponding to the inputs X_* can be sampled from the posterior distribution to estimate $f(\mathbf{x})$. This provides the main basis for GPR.

FIC Approximation

The approximation method for GPR I used was the Fully Independent Conditional (FIC) Approximation. This approximation method modfies the SoR method in a way that avoids the degeneracy problem as in the SoR method, so it is operating on a GP still. In FIC, only a subset of the full dataset is used, and depending on the data being inputted into the kernel, it may either be approximated or calculated exactly. The kernel for the FIC approximation is shown below:

$$\begin{aligned} \bar{k}_{\text{FIC}}(x_i, x_j \mid \theta, A) = (1 - \delta_{ij})\bar{k}_{\text{SR}}(x_i, x_j \mid \theta, A) \\ +\delta_{ij}k(x_i, x_j \mid \theta, A) \end{aligned}$$

As seen above, the kernel is approximated when the $i \neq j$, which is equivalent to the SoR method, and calculated exactly when $i = j$. The FIC approximation of the $K(X, X \mid \theta, A)$ matrix is given as:

$$\begin{aligned} \bar{K}_{\text{FIC}}(X, X \mid \theta, A) = \bar{K}_{\text{SR}}(X, X \mid \theta, A) \\ +\delta_{ij}(k(x_i, x_j \mid \theta, A)) \end{aligned}$$

This method was chosen over the other approximation methods for a number of reasons. This method was chosen over the SoR Method due to FIC being a direct improvement to it, as it has the same computational complexity as the SoR while fixing the issue with the degeneracy of the covariance matrix. The PP Approximation was not used due to reasons mentioned earlier, and the BCM Method was not chosen due to it's higher computational complexity for similar performance. The last method that was not chosen was the SoD method, and while it does have the same computational complexity, it is based on a degenerate GP, so I chose FIC over it for the same reasons as SoR. Overall, the FIC Approximation provided a strong middle ground of the different approximation methods, leading to it being chosen.

Analysis and Results

Overall, as the model progressed through the joint torques the MSE substantially decreased. This could possibly be due to the fact that earlier joints in the system (assuming earlier indexed joint variables are closer to the base as is standard) have a more dramatic effect on the location of the end effector, as small changes in q_1 or q_2 could cause large changes in the position, velocity, and acceleration of the arm, especially if the arm is large. The extremely low MSE for the later joints indicates that GPR performed well in terms of predicting their respective torque functions. It is possible that if I were to fine-tune the optimization of the hyperparemeters or define custom kernels that I could have achieved better performance on the earlier joint torques. Overall, the GPR seems to be able to well predict the torque variables for the SARCOS robotic arm.

Table 1:Details the MSE, kernel function, and basis function for each torque variable (denoted q_i) estimated.

i	MSE for predicted q_i	q_i Kernel	q_i Basis
1	12.7200	R. Quad.	Pure Quad.
2	5.0697	Matern52	Constant
3	1.3051	Matern52	Linear
4	0.8692	Matern32	Quad.
5	0.0271	Exponential	None
6	0.3276	R. Quad.	Pure Quad.
7	0.0962	R. Quad.	Pure Quad.

Conclusion

Using GPR, I was able to predict the functions of each torque of the arm using the position, velocity, and acceleration of each joint. GPR was better able to predict joints further from the base of the robot; future analysis could be performed in an attempt to provide stronger predictions for the earlier torques, possibly by defining a custom kernel for them or altering how the hyperparameters are optimized.