Charlie meeting 20th June

Meeting

In this meeting, Andrew and I discussed the result I produced last week. Everything is working, which is good. This week I will work on different type of paramtisation to paramatise the energy function, Andrew suggested a few different methods as basis functions, such as polynomial, splines and Fourier representation. We also talked about my methods of adding jitter, which might not be entire reasonable so I am fixing that too. He also talked about a way that makes this methods scalable, which is that instead of a grid of invariance, we only condition on the local area around the data points as well as the test points we wish to evaluate. If noise too small or too much data, it's not good for nonlinear nor damping. But in most reasonable cases, 30 seconds with 30 points with noise 0.1 the kernel all worked well. We have four scenarios to check. Noisy being 0.1 and not noisy being 0.01. Lots of data means 0.1 time spacing. left is naive, right is invariance

1. lots of data, noisy
2. little data, noisy
3. lots of data, not noisy
4. little, not noisy

	lots of data, noisy	little data, noisy	lots of data, not noisy	little data, not noisy
SHM	-1323, -1305	65, 81	1354, 1394	271, 311
Fixed Damped SHM	-1310, -1292; -1305, -1290; -1310, -1297	78, 96; 79, 92; 54, 69	1387, 1419; 1349, 1338; 1404, 1346	289, 322; 269, 254; 329, 253
Dynamical Damped SHM	-1310, -1292; -1305, -1289; -1310, -1295	78, 95; 79, 96; 54, 68	1387, 1423; 1349, 1382; 1404, 1435	289, 316; 269, 258; 329, 256
Pendulum	-1307, -1288	61, 65	1403, 1451	232, 115
Fixed Damped Pendulum	-1257, -1239; -1324, -1308; -1315, -1301	52, 62; 55, 69; 67, 79	1397, 1428; 1401, 1363; 1373, 1266	272, 133; 267, 179; 296, 175
Dynamical Damped Pendulum	-1257, -1239; -1324, -1308; -1315, -1299	52, 62; 55, 70; 67, 82	1397, 1442; 1401, 1442; 1373, 1412	272, 135; 267, 209; 296, 241

We can see that most of the entries are good, except lots of data not noisy for fixed damping when damping is high, but dynamical damping still outperforms. A lot of problem is found in low data, low noise area; especially when it is non linear or damping is high. This make sense in some way. For the first case, since it is harder to for fixed damping to handle and predict larger damping, if there are less noise to smooth out the effect of damping, so that damping is more varying, it makes sense for fixed damping to perform worse. For the second case, I think the main reason is that the estimation of acceleration and velocity is already not so good to start with, if we have low noise, it's like we are assuming that's the truth. Therefore, our physical intuition (invariance) will misguide us since we will be expecting something else.

Paramatisation

Local Invariance

Before we have

$$\begin{pmatrix} \mathbf{f}(X_1) \\ \mathbf{f}(X_2) \\ L_{X_g}[\mathbf{f}(X_g)] \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mathbf{0}_{2n} \\ \mathbf{0}_{2m} \\ \mathbf{0}_{\ell} \end{pmatrix}, \begin{pmatrix} K(X_1, X_1) & K(X_1, X_2) & B_1 \\ K(X_2, X_1) & K(X_2, X_2) & B_2 \\ C_1 & C_2 & D \end{pmatrix} \right),$$

Before with the grid of invariance X_g is fixed, now they are consist of the local area around the test points and the training trajectory data. Therefore, the only change we need to make is X_g is now a function of X_1, X_2 so that $X_g = X_g(X_1, X_2)$. As a result, now B, C, D will all be a function of X_1, X_2. Essentially, we just have to redefine our grids to be the neighbourhoods around test and training points. In our case, before we have 1600 points in a 6 by 6 area, now perhaps we can have 4 points around the 0.1 by 0.1 area around each point.

Change to damping approximate invariance

Now instead of our original method of conditioning on an either analytical form or like a fix or random vector, we will allow the invariance to be noisy. Before we had,

$$\begin{pmatrix}\mathbf{f}(X_1)\\ \mathbf{f}(X_2)\end{pmatrix} | L_{X_g}[\mathbf{f}(X_g)] = 0 \sim \mathcal{N}\left(\begin{pmatrix}\mathbf{0}_{2n}\\ \mathbf{0}_{2m}\end{pmatrix}, A - \begin{pmatrix}B_1\\ B_2\end{pmatrix} D^{-1}\left(C_1, C_2\right)\right)$$

now we want

$$\begin{pmatrix}\mathbf{f}(X_1)\\ \mathbf{f}(X_2)\end{pmatrix} | L_{X_g}[\mathbf{f}(X_g)] = \text{noise} \sim \mathcal{N}\left(\begin{pmatrix}\mathbf{0}_{2n}\\ \mathbf{0}_{2m}\end{pmatrix}, A - \begin{pmatrix}B_1\\ B_2\end{pmatrix} D^{-1}\left(C_1, C_2\right)\right)$$

We had

$$\begin{pmatrix}\mathbf{f}(X)\\ L_{X_g}[\mathbf{f}(X_g)]\end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix}\mathbf{0}_{2n}\\ \mathbf{0}_{\ell}\end{pmatrix}, \begin{pmatrix}A & B\\ C & D\end{pmatrix}\right),$$

Question

I still don't quite understand the difference between conditioning on a random vector and add noise to the thing itself.