1 Current

1.1 Using SDEs to approximate the output distribution of a model

- How to relate the linear SDE with the sequence y produced by N? Here is a proposition. The model N already provides the mean μ and the diagonal of the covariance matrix (σ^i). We need to find coefficients a, b, c, an initial condition y_0, and a time step τ such that the sampled y has its mean and covariance diagonal as close as possible to μ and (σ^i).
- Is any Gaussian vector a sample of an OU process? Probably not because the sample of an OU process is determined by 5 parameters (a, b, c, y_0, τ), while a Gaussian vector has $m(m+3)/2$ parameters. If we allow the time step to vary, we obtain $m+4$ parameters $(a, b, c, y_0, \tau_1, \dots, \tau_m)$, so we can fix the mean (or the covariance diagonal) but there are still $m(m+1)/2 - 4$ degrees of freedom.

1.2 Visualization

- Use 1-dimensional dynamical systems: simple nonlinear system, linear system, and linear system with a small nonlinear perturbation. The leaves can be plotted in a 3-dimensional input space.
- I may also try 2-dimensional dynamical system for more interesting behavior. However, the Takens condition cannot be verified since the input space must have at least 5 dimensions while I am limited to 3 for visualization. So I can only hope that the Takens map is still an embedding.
- How can we explain the "separation line" between positive and negative values?
- In order to investigate the relation between the model and the true dynamical system, it may be relevant to generate a simulated dataset in the following manner. We choose a manifold of "high" dimension ($n \sim 10$) and a diffeomorphism on this manifold as our true dynamical system. We choose a smooth measurement function sending any point from the manifold (a state) to a real number. The model is trained on sufficiently long sequences of measurements (at least $2n+1$). The problem with this idea is that we cannot plot the leaves since they have more than three dimensions. But once we obtain a precise result on the kernel foliation, we should evaluate it using this idea.
- For example, the true dynamical system can be a simplified version of the flight equations of an A320 ($n = 12$) and the measurement function can be the altitude. If the input space is the set of sequences of altitude measurements of length $2n+1$, it is possible to plot the trajectories in this space using the true dynamical system. The trajectories will lie on a submanifold of dimension n. The foliation induced by the model can then be plotted in the same space.
- We can also use OpenSky data for the same purpose.

1.3 General concepts

- The Quest for a General Theory of Robustness in Learning Machines.
- Concerning the sequential models, it may be relevant to extend the definition to include sequences of vectors (instead of sequences of scalars only)
- The definition of sequential models should also include models that process sequences of variable length.
- Can we obtain a criteria on the length of the output sequence for the robustness?

2 Old

2.1 Visualization

- The chaotic behavior of some kernel leaves may be caused by a standard deviation $\sigma = 0$ (capped at 10^{-6}) in which case the kernel of the local data matrix may have dimension 0, or 2, or 3, or be undefined.

The chaotic behavior is indeed due to $\sigma = 0$. In this case, the local data matrix should be undefined since the FIM of the output space is undefined. But, since I capped σ to a small nonzero value, I do not get any error. However, the leaves thus obtained are irrelevant. I need to find a way to force the network to output nonzero σ.

- Check the computation of the local data matrix. It may not be necessary to compute the FIM on the output space since we can directly differentiate the negative log-likelihood (in fact, both approaches are certainly equivalent).

  If you differentiate the negative log-likelihood, you still need to compute the expectation with respect to the output distribution. As far as I know, there is no obvious analytical formula for doing this with the normal distribution. It can be done numerically but I want to avoid that. So the first method of computing the FIM on the output space, then pullback it to the input space is the best one IMO.

- Force the network to output nonzero standard deviation.

  This was done with $\sigma = y^2$ instead of $\sigma = \mathrm{ReLU}(y)$. We may also test $\sigma = \exp(y)$. This significantly increases the likelihood. Moreover, the likelihood finally behaves similarly as the absolute error. However, this seems very sensitive to the capping for the loss function. First, I used $\epsilon = 1e - 100$, then $\epsilon = -\infty$, and the likelihood was higher in the former than in the later, which is counter-intuitive. It may be due to the initialization of the model, so I should run several training sessions.