Neural Particle Filters

Jean-Pascal Pfister

Anna Kutschireiter  Simone Surace  Henning Sprekeler

Bernstein workshop: Neural Sampling
• The filtering task
  ▸ formalisation and desiderata

• The Bootstrap Particle Filter
  ▸ curse of dimensionality

• The Neural Particle Filter
  ▸ inference and learning
  ▸ Avoiding the curse of dimensionality

• The spiking Neural Particle Filter
Outline

• The filtering task
  ‣ formalisation and desiderata

• The Bootstrap Particle Filter
  ‣ curse of dimensionality

• The Neural Particle Filter
  ‣ inference and learning
  ‣ Avoiding the curse of dimensionality

• The spiking Neural Particle Filter
Mathematical formulation of the problem

Generative model

Hidden process (e.g. the position of the prey):

\[ dx_t = f(x_t) dt + \Sigma_x^{1/2} dw_t \]

Observation process (e.g. visual stimulus):

\[ dy_t = g(x_t) dt + \Sigma_y^{1/2} dv_t \]

Question: \( p(x_t | Y_t) = ? \)

What is the posterior distribution over the hidden causes \( x_t \) given the past observations \( Y_t = \{ y_s, s \leq t \} \)?
Desiderata for the filter

➡ Can represent dynamic distributions
➡ Can represent arbitrary distributions
➡ Works with nonlinear generative models
➡ is scalable with dimensions
➡ is learnable online
➡ is implementable in a neural circuit
➡ is robust to neuron pruning
Outline

• The filtering task
  ▸ formalisation and desiderata

• The Bootstrap Particle Filter
  ▸ curse of dimensionality

• The Neural Particle Filter
  ▸ inference and learning
  ▸ Avoiding the curse of dimensionality

• The spiking Neural Particle Filter
The Bootstrap Particle Filter

**Particle representation**

\[ p(x_t|y_0...t) = \sum_{i=1}^{N} w_t^i \delta(x_t - \hat{x}_t^i) \]

**Weight update**

\[ w_t^i \propto w_{t-\Delta t}^i p(dy_t|\hat{x}_t^i) \]

**Particle dynamic**

\[ d\hat{x}_t^i = f(\hat{x}_t^i)dt + \sum_{i}^{1/2} d\hat{w}_t^i \]

**Problem 1: Biological implementation ??**
Problem 2: collapse of the # of effective particles

\[ N_t^{\text{eff}} = \left( \sum_i (w_t^i)^2 \right)^{-1} \rightarrow 0 \]
Problem 2: collapse of the # of effective particles

\[ dN_{\text{eff}}^t \propto -DN_{\text{eff}}^t \, dt \]

\[ D = \text{dim}(x) = \text{dim}(y) \]

![Graph showing the collapse of the effective sample size over time for different dimensions and ensemble sizes.](image)

\[ T(D, N, N_{\text{eff}}^t) \]

Surace et al. ArXiv. 2017
Problem 2: collapse of the # of effective particles

\[ dN_{t}^{\text{eff}} \propto -DN_{t}^{\text{eff}} \, dt \]

Figure 1: Top left: The collapse of the effective sample size \( \tilde{N}_{t}^{\text{eff}} \) over time is shown for different dimensions \( D \) of the state space and for an ensemble size \( N = 10^{4} \). The displayed time-course is an average of 100 independent trials.

Top right: A plot to the stopping time \( T(D,N,n) \) for \( n = 10 \) shows an approximate \( D^{-1} \) scaling (thick black line).

Bottom: The time evolution of the mean squared error shows adiabatic and subsequent deterioration due to weight decay for dimensions up to 50. The dip is shallower and the deterioration faster for higher dimensions. For even higher dimensions, the dip is no longer visible because the weight decay is too quick. All traces are averages of 100 independent trials with an ensemble size of \( N = 10^{4} \).
What about resampling?

resampling time
Resampling helps only after a certain delay

The benefit of resampling is that particles with vanishing weights are discarded, whereas the time-scale of weight degeneracy continues to decay with time (thick black line, D = 100). After resampling, the immediate implication is that with a fixed ensemble size, the initial magnitude of change of the empirical variance is of the same order of magnitude as the (true) posterior standard deviation, which is proportional to \( \frac{1}{\tau_{\text{MSE}}} \), where \( \tau_{\text{MSE}} \) is the inverse of the sum of the squared weights, is also proportional to \( \sqrt{N} \), where \( N \) is the ensemble size.

Resampling temporarily improves filter performance relative to the initial decay. The immediate effect of resampling for an ensemble size of \( N \approx 50 \) is around half of that for an initialization from the prior. However, since resampling does not add any new information about the true state, the benefit of resampling is that particles with vanishing weights are discarded, whereas the time-scale of weight degeneracy continues to decay with time. This confirms that the scaling is close to proportional to \( \frac{1}{\sqrt{D}} \), for which we rely on a numerical investigation. The results are shown in Fig. 3, D, N, n.

We measure our time-course in the left panel. The characteristic time of one time unit.

Despite the immediate performance increase, due to the decay of the empirical variance, it is therefore expected that resampling leads to an immediate performance increase. This is because of the decay of the empirical variance, which is of the same order of magnitude as the (true) posterior standard deviation. The time that is needed to reach such a state is related to the inverse of the sum of the squared weights, and is high, the slope of the linear fits to the initial segment of the MSE time-course in the left panel shows the slope of the empirical variance.

The empirical variance is of the same order of magnitude as the (true) posterior standard deviation, which is proportional to \( \frac{1}{\tau_{\text{MSE}}} \), where \( \tau_{\text{MSE}} \) is the inverse of the sum of the squared weights, is also proportional to \( \sqrt{N} \), where \( N \) is the ensemble size. Thus the initial magnitude of change of the empirical variance is of the same order of magnitude as the (true) posterior standard deviation, which is proportional to \( \frac{1}{\sqrt{D}} \), for which we rely on a numerical investigation. The results are shown in Fig. 3, D, N, n.

In order to quantify the time-scale of the MSE decay just after resampling, we measure our time-course in the left panel. The characteristic time of one time unit.

Next, we study the dependence of the benefit of resampling on the number of new data points per time unit.

Since resampling does not add any new information about the true state, the benefit of resampling is that particles with vanishing weights are discarded, whereas the time-scale of weight degeneracy continues to decay with time. This confirms that the scaling is close to proportional to \( \frac{1}{\sqrt{D}} \), for which we rely on a numerical investigation. The results are shown in Fig. 3, D, N, n.

In Fig. 3, D, N, n, we numerically estimate the scaling by using trial averages, and we show the results as a function of different values of \( D \).

The time that is needed to reach such a state is related to the inverse of the sum of the squared weights, and is high, the slope of the linear fits to the initial segment of the MSE time-course in the left panel shows the slope of the empirical variance.

The empirical variance is of the same order of magnitude as the (true) posterior standard deviation, which is proportional to \( \frac{1}{\sqrt{D}} \), for which we rely on a numerical investigation. The results are shown in Fig. 3, D, N, n.

The time that is needed to reach such a state is related to the inverse of the sum of the squared weights, and is high, the slope of the linear fits to the initial segment of the MSE time-course in the left panel shows the slope of the empirical variance.

The empirical variance is of the same order of magnitude as the (true) posterior standard deviation, which is proportional to \( \frac{1}{\sqrt{D}} \), for which we rely on a numerical investigation. The results are shown in Fig. 3, D, N, n.

The empirical variance is of the same order of magnitude as the (true) posterior standard deviation, which is proportional to \( \frac{1}{\sqrt{D}} \), for which we rely on a numerical investigation. The results are shown in Fig. 3, D, N, n.

The empirical variance is of the same order of magnitude as the (true) posterior standard deviation, which is proportional to \( \frac{1}{\sqrt{D}} \), for which we rely on a numerical investigation. The results are shown in Fig. 3, D, N, n.
Resampling does not help

Resampling temporarily improves filter performance relative to initial decay. The benefit of resampling is that particles with vanishing weights are discarded, whereas the time-scale of weight degeneracy co-occurs. The immediate implication is that with a fixed ensemble size, particles are located at positions where the likelihood of observation is proportional to the empirical variance, which is of the same order of magnitude as the (true) posterior variance. The particles have to disperse away from resampled positions. For example, in the extreme case where due to resampling does not help lead to an immediate performance increase.

We measure weight degeneracy. The immediate implication is that with a fixed ensemble size, the time that is needed to reach such a state is related to the inverse of the sum of the squared weights, which is the inverse absolute value of the slope, 

\[ \tau_{MSE} \propto D_{crit} \]

This confirms that the scaling is close to 

\[ N \approx D \]

This manuscript is for review purposes only.

In order to quantify the time-scale of the MSE decay just after resampling, we numerically estimate \( T_{crit} \), which is the inverse of the sum of the squared weights, is also proportional to the deviation, which is equal to 

\[ \frac{1}{T_{MSE}} \]

We measure the characteristic time for the initial segment of the MSE time-course in the left panel. The dip in the time-course of MSE shows a delayed effect of resampling.

After resampling, the thin straight lines show the slope of \( \tau_{MSE} \) performance as shown by the dip in the time-course of MSE.

The slopes of the linear fits to the initial segment of the MSE time-course continue to decay with time, whereas the time-scale of weight degeneracy continues to decay with time.

If the criterion to resample the particles is that the initial magnitude of change of the posterior variance is of the same order of magnitude as the true posterior standard deviation, which is the inverse of the sum of the squared weights, is also proportional to \( \frac{1}{T_{MSE}} \).

Thus the initial magnitude of change of the posterior variance is of the same order of magnitude as the (true) posterior standard deviation, which is the inverse of the sum of the squared weights, is also proportional to the deviation, which is equal to 

\[ \frac{1}{T_{MSE}} \]

We numerically estimate \( T_{crit} \), which is the inverse of the sum of the squared weights, is also proportional to the deviation, which is equal to 

\[ \frac{1}{T_{MSE}} \]

We numerically estimate \( T_{crit} \), which is the inverse of the sum of the squared weights, is also proportional to the deviation, which is equal to 

\[ \frac{1}{T_{MSE}} \]

We numerically estimate \( T_{crit} \), which is the inverse of the sum of the squared weights, is also proportional to the deviation, which is equal to 

\[ \frac{1}{T_{MSE}} \]

We numerically estimate \( T_{crit} \), which is the inverse of the sum of the squared weights, is also proportional to the deviation, which is equal to 

\[ \frac{1}{T_{MSE}} \]
The curse of dimensionality of the BPF can be seen as a point estimate of the state, which becomes exact for very small observation noise. In higher dimensions, the number of particles needed for a standard (weighted) particle filtering algorithm grows exponentially with the number of dimensions, i.e. an algorithm suffering from the 'curse of dimensionality'. The unweighted approaches avoid the 'curse of dimensionality'. This is particularly suited to model perception phenomenologically, because it shares some important properties considered crucial for perception. First, perception relies on noisy and incomplete sensory data, as for instance encountered in visual scenes, and uses these to make sense of the world, which in our model is represented by a high-dimensional hidden state variable. Second, the brain needs to combine information from different sensory cues efficiently in order to decrease uncertainty or ambiguity. In addition, it exploits strong nonlinearities: the hidden state variable, which in our model is reparameterized by a weightless linear generative model, is further implementable as a neuronal dynamics in a recurrent neuronal network. It may thus serve as a step towards understanding how perception can be implemented in the brain on a conceptual level.

In this paper, we set perception in the context of the computational task of nonlinear Bayesian filtering, we proposed an analog dynamics for particles (or neurons) that serves as a filtering approach such as the NPF. Of course, any filtering algorithm employed by a neuronal population for perception should be economical in its resources: an algorithm that needs an exponential amount of neurons with growing dimension, i.e. an algorithm suffering from the 'curse of dimensionality', would be devastating. Instead, the number of particles scales linearly with hidden dimensions. Solid lines correspond to linear (in higher dimensions). The large number of hidden variables, a requirement that is fulfilled by the theory of nonlinear filtering, we proposed an analog dynamics for particles (or neurons) that serves as a filtering approach such as the NPF. Of course, any filtering algorithm employed by a neuronal population for perception should be economical in its resources: an algorithm that needs an exponential amount of neurons with growing dimension, i.e. an algorithm suffering from the 'curse of dimensionality', would be devastating. Instead, the number of particles scales linearly with hidden dimensions. Solid lines correspond to linear (in higher dimensions). Contrarily, the unweighted approaches avoid the 'curse of dimensionality' and the number of particles scales linearly with hidden dimensions.
The curse of dimensionality of the BPF
• The filtering task
  ▸ formalisation and desiderata

• The Bootstrap Particle Filter
  ▸ curse of dimensionality

• The Neural Particle Filter
  ▸ inference and learning
  ▸ Avoiding the curse of dimensionality

• The spiking Neural Particle Filter
 Particle representation

\[ p(x_t|y_0...t) = \sum_{i=1}^{N} w_t^i \delta(x_t - \hat{x}_t^i) \]

weight update

\[ w_t^i \propto w_{t-dt}^i p(dy_t|\hat{x}_t^i) \]

Particle dynamic

\[ d\hat{x}_t^i = f(\hat{x}_t^i)dt + \Sigma_1^{1/2}d\hat{w}_t^i \]
The Neural Particle Filter

Particle representation

\[ p(x_t | y_0...t) = \sum_{i=1}^{N} w_t^i \delta(x_t - \hat{x}_t^i) \]

no weight update

\[ w_t^i = \frac{1}{N} \]

Particle dynamic

\[ d\hat{x}_t^i = f(\hat{x}_t^i)dt + W_t(dy_t - g(\hat{x}_t^i)dt) + \Sigma_x^{1/2} d\hat{w}_t^i \]

“Kalman” gain:

\[ W_t = \text{cov}(x_t, g(x_t)) \]
The Neural Particle Filter

Hidden process:
\[ dx_t = f(x_t)dt + \Sigma_{x}^{1/2}dw_t \]

Observation process:
\[ dy_t = g(x_t)dt + \Sigma_{y}^{1/2}dv_t \]

Neural Particle Filter:
\[ d\hat{x}_t = f(\hat{x}_t)dt + W(dy_t - g(\hat{x}_t)dt) + \Sigma_{x}^{1/2}d\hat{w}_t \]

posterior distribution
Implementation of the Neural Particle Filter

Hidden process:
\[ dx_t = f(x_t)dt + \Sigma_x^{1/2}dw_t \]

Observation process:
\[ dy_t = g(x_t)dt + \Sigma_y^{1/2}dv_t \]

Neural Particle Filter:
\[ d\hat{x}_t = f(\hat{x}_t)dt + W(dy_t - g(\hat{x}_t)dt) + \Sigma_x^{1/2}d\hat{w}_t \]

\[ y_t \rightarrow \hat{n}_t \rightarrow \hat{x}_t \rightarrow W \rightarrow f(\hat{x}_t) \]

- *input neurons*
- *novelty neurons*
- *filter neurons*
Parameter learning

**Innovation process:**

\[ dn_t = dy_t - \langle g_\theta(x_t) \rangle_{p_\theta(x_t | \mathcal{Y}_t)} dt \]

**Prediction error**

\[ dy_t = \langle g(x_t) \rangle_{p_\theta(x_t | \mathcal{Y}_t)} + dn_t \]

\[ dy_t = dw_t \]

\[ \Lambda_\theta(\mathcal{Y}_t) = \frac{dQ_\theta(\mathcal{Y}_t)}{dP(\mathcal{Y}_t)} = \exp \left( \int_0^t \langle g_\theta(x_s) \rangle^T dy_s - \frac{1}{2} \langle g_\theta(x_s) \rangle^T \langle g_\theta(x_s) \rangle ds \right) \]

**Wiener process**

\[ \text{Girsanov theorem} \]

\[ \langle \cdot \rangle_{p_\theta(x_s | \mathcal{Y}_s)} \]

**Radon-Nikodym derivative**

Bain and Crisan, 2009  Surace and Pfister, 2017
Parameter learning

Informal: same as minimize the prediction error:

\[ \dot{\theta} = \frac{\eta}{2dt} \nabla_{\theta} \| dy_t - \langle g_{\theta}(x_t) \rangle_{p_{\theta}(x_t|y_t)} dt \|^2 \]

Online Learning rule: by gradient ascent on \( d \log \Lambda_{\theta}(Y_t) \)

\[ \dot{\theta} = \eta (\nabla_{\theta} \langle g_{\theta}(x_t) \rangle^T) (dy_t - \langle g_{\theta}(x_t) \rangle dt) \]

\[ \Lambda_{\theta}(Y_t) = \frac{dQ_{\theta}(Y_t)}{dP(Y_t)} = \exp \left( \int_0^t \langle g_{\theta}(x_s) \rangle^T dy_s - \frac{1}{2} \langle g_{\theta}(x_s) \rangle^T \langle g_{\theta}(x_s) \rangle ds \right) \]
Position of the prey:
\[ dx_t = 3x_t(1 - x_t^2)\ dt + d\omega_t, \]

Visual input:
\[ dv_t = x_t\ dt + \sigma_v d\beta_t, \]

Auditory input:
\[ da_t = \tanh(2x_t)\ dt + \sigma_a d\gamma_t \]

Neural Particle Filter:
\[ d\hat{x}_t = 3\hat{x}_t(1 - \hat{x}_t^2)\ dt + d\omega_t + W_t^{(v)}(dv_t - \hat{x}_t\ dt) + W_t^{(a)}(da_t - \tanh(2\hat{x}_t)\ dt) \]
Figure 1: The NPF as a model for perception for multisensory perception.

(a) The frog has to do filtering in order to dynamically track the position $x_t$ of the insect, which is switching between two branches. We model this behavior by the stochastic dynamics in Eq. (6), which gives rise to a bimodal prior distribution $p(x_t = x)$. In order to track the insect, the frog has to make use of this prior, and optimally combine it with the noisy input from the sensory modalities $v_t$ and $a_t$.

(b) Tracking simulation with $N = 1000$ filtering neurons and sensory noise $\sigma^2_v = 0.1, \sigma^2_a = 1$. The upper panel shows the true trajectory of the insect (solid black line) and particle densities. The regions between the dotted black lines denote the two branches, and certainty levels certainty levels in the middle panel correspond to the relative number of particles whose states are within one of the two branches. At each time, the sensory gains in the lower panel are computed according to Eqs. (10) and (11).

(c) Performance in terms of time-averaged mean-squared error $\text{MSE} = \langle (x_t - \hat{x}_t)^2 \rangle$ (dotted lines: standard particle filter) and time-averaged gains $\langle W_t(v_t) \rangle$ (solid line) and $\langle W_t(a_t) \rangle$ (dashed line) as function of sensory noise.

This explains why the predictive performance of the EKF in terms of MSE is fairly poor compared to that of the NPF (Fig. 2a,b). For larger observation noise, matters become even worse because the state estimate of the EKF evolves to one of the fixed points of $f(x)$ and remains there, irrespective of the real hidden state.
within one of the two branches. At each time, the sensory gains in the lower panel are
which is switching between two branches. We model this behavior by the stochastic
noise.

time-averaged gains
mean-squared error
levels in the middle panel correspond to the relative number of particles whose states are
between the dotted black lines denote the two branches, and certainty levels certainty
to track the insect, the frog has to make use of this prior, and optimally combine it with
dynamics in Eq.

Figure 1:

For larger observation noise, matters become even worse because the state estimate of the EKF


Performance in terms of time-averaged

\( MSE \)

\( \langle W_{t}^{(v)} \rangle \)

\( \langle W_{t}^{(a)} \rangle \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)

\( \sigma_{v}^{2} \)

\( \sigma_{a}^{2} \)
weights are not local, i.e. they rely on the state of the whole network (cf. Eqs. 22 and 24). However, in the deterministic limit the learning rule for the generative weight $J$ can be replaced by a learning rule that is both Hebbian and local and relies on a multiplication between pre- and postsynaptic activity (Eq. 12 and, more generally, Eq. 6).

In general, the learning rules for these generative weight parameters, learning starts at the Hebbian learning rule (12) in Eq. (5). As $\sigma_v^2$ approaches the true value, the MSE of the NPF resembles that of the standard PF.

The filtering performance is not affected by the bias of estimated $\hat{J}$.

**Hebbian learning rule**

$$\dot{\theta} = \eta(\nabla_{\theta} \langle g_\theta(x_t) \rangle^T)(dy_t - \langle g_\theta(x_t) \rangle dt)$$

$$g_\theta(x) = Jx$$

small $\Sigma_y$

$$\dot{J} \approx \eta(dy_t - Jx_t dt)x_t^T$$

**ML learning vs Hebbian learning**

Simulations shown here correspond to the example model with Eqs. (a), (b), and (c). As $\sigma_v^2$ approaches the true value, the MSE of the NPF resembles that of the standard PF.

For sensory noise $\sigma_x$, which is a valid approximation for small sensory noise. As $\sigma_v^2$ approaches the true value, the MSE of the NPF resembles that of the standard PF.

NPF avoids the curse of dimensionality

![Graph showing the number of particles needed versus hidden dimensionality for PF and NPF](image-url)

- **PF** (Particle Filter) shows an exponential increase in the number of particles required as the hidden dimensionality increases.
- **NPF** (Normalized Particle Filter) avoids the 'curse of dimensionality', indicating a more efficient approach.

**Discussion**

- Perception relies on noisy and incomplete sensory data, similar to how the brain processes information.
- The brain can be seen as a point estimate of the state, which becomes exact for very small observation noise.
- Trajectories of the samples are observed, and in the unweighted approaches, each particle state itself can be seen as a point estimate of the state.

**Besides being just an algorithmic trait, the scaling with dimensions does have biological relevance.** Consider different sensory cues expected by the brain. The number of hidden dimensions required increases as the hidden state space becomes more complex.

**Perception should be economical in its resources:** An algorithm that needs an exponential amount of resources to solve the problem of inferring the hidden state is not optimal.

**In higher dimensions, the unweighted approaches NPF and FBPF outperform the (standard) weighted particle filter.** The number of particles scales linearly with hidden dimensions. Solid lines correspond to linear filtering, whereas dashed lines represent exponential scaling.

**Figure 5**

- **PF** (Particle Filter) least-squares filtering performance in terms of MSE (normalized) is particularly suited to model perception phenomenologically.
- **NPF** both inherently avoids the 'curse of dimensionality' and offers an analog dynamics for particles (or neurons) that serves as a step towards understanding how perception can be implemented in the brain on a conceptual level.
NPF avoids the curse of dimensionality
Outline

- The filtering task
  - formalisation and desiderata
- The Bootstrap Particle Filter
  - curse of dimensionality
- The Neural Particle Filter
  - inference and learning
  - Avoiding the curse of dimensionality
- The spiking Neural Particle Filter
Spiking Neural Particle Filter (sNPF)

Generative model

Hidden process:
\[ dx_t = f(x_t)dt + \Sigma_x^{1/2}dw_t \]

Observation process:
\[ d\hat{N} \sim \text{Poisson}(g(x)dt) \]

Spiking Neural Particle Filter

\[ d\hat{x}_t = f(\hat{x}_t)dt + W(d\hat{N}_t - g(\hat{x}_t)dt) + \Sigma_x^{1/2}d\hat{w}_t \]

“Kalman” gain:
\[ W_t = \frac{\text{cov}(g(x), x)}{\langle g(x) \rangle} \]

Kutschireiter et al. Bernstein conference poster
The sNPF nicely tracks the stimulus

Figure 1: Left: The filter estimate (orange) is able to track the hidden stimulus (black line, nonlinear hidden dynamics \( f(x) \)/\( x(x^2+1) \), given grayscale distribution) based on a spike train of 50 neurons. Dark shading corresponds to regions of high particle density and thus high posterior density. In this simulation, we used an exponential rate function that takes into account refractoriness via an explicit time dependence. Right: Number of particles \( N \) needed to achieve a fixed filtering performance with increasing hidden dimensionality scales linearly for the sNPF (dashed line: linear fit), and exponentially for a weighted particle filter (dashed line: exponential fit) for a multivariate generalization of the nonlinear toy model in the left panel.
The Spiking Neural Particle Filter avoids the COD

Number of particles required to reach MSE = 0.8 var(prior)

![Graph showing the number of particles required for different dimensionality](image-url)
### Summary: comparison of filters

<table>
<thead>
<tr>
<th>Feature</th>
<th>KF</th>
<th>BPF</th>
<th>NPF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can represent dynamic distributions</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Can represent arbitrary distributions</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Works with nonlinear gen. models</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Is scalable with dimensions</td>
<td>✓</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Is learnable online</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Is implementable in a neural circuit</td>
<td>?</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Is robust to neuron pruning</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Acknowledgements and bibliography

Surace, Kutschireiter and Pfister, *How to avoid the curse of dimensionality: scalability of particle filters with and without importance weights*, *ArXiv*, 2017

