A Discriminative Approach to Bayesian Filtering with Applications to Human Neural Decoding

Michael C. Burkhart

Division of Applied Mathematics
Brown University
Providence, Rhode Island

23 May 2018

Overview

1. Bayesian Filtering
   - problem description
   - main approaches
   - applications
2. Challenges to Effective Neural Decoding
   - what makes this problem hard / unique
   - drawbacks to current approaches
   - our solution: the DKF
3. Nonstationarities
   - the problem and proposed solutions
   - our approach

State-Space Models

Consider the following graphical model:

$$\begin{tikzcd} Z_1 \arrow[r] \arrow[d] & \cdots \arrow[r] & Z_{t-1} \arrow[r] \arrow[d] & Z_t \arrow[d] & \} \text{ hidden} \\ X_1 & & X_{t-1} & X_t & \} \text{ observed} \end{tikzcd}$$

Filtering calculates our best guess for Z_t given the sequence of observations $X_1, \ldots, X_t$.

Bayesian Filtering

In the *Bayesian* approach to filtering, we place a joint distribution on these quantities:

- The *state model* $p(z_t|z_{t-1})$ relates the current and previous hidden states.
- The *measurement model* $p(x_t|z_t)$ relates the current observation and hidden state.

$$\begin{tikzcd}
Z_1 \arrow[r, "p(z_2|z_1)"] \arrow[d, "p(x_1|z_1)"] & \cdots \arrow[r, "p(z_{t-1}|z_{t-2})"] & Z_{t-1} \arrow[r, "p(z_t|z_{t-1})"] \arrow[d, "p(x_{t-1}|z_{t-1})"] & Z_t \arrow[d, "p(x_t|z_t)"] \\
X_1 & & X_{t-1} & X_t
\end{tikzcd}$$

The Posterior

We are interested in the posterior $p(z_t|x_{1:t})$.

This can be calculated recursively via:

$$p(z_t|x_{1:t}) \propto \underbrace{p(x_t|z_t) \underbrace{\int p(z_t|z_{t-1})\, p(z_{t-1}|x_{1:t-1})\, dz_{t-1}}_{\text{state update, gives } p(z_t|x_{1:t-1})}}_{\text{measurement update, gives } p(x_t, z_t|x_{1:t-1})}$$

The proportionality entails dividing by $p(x_t|x_{1:t-1})$.

Methodology

Approaches to Bayesian filtering mirror approaches to integration:

- **exact**: the Kalman filter [Kal60; KB61], extended by Beneš [Ben81] and Daum [Dau84; Dau86]
- **variational inference**: Extended Kalman Filter (EKF), Laplace approximation, Gaussian Assumed Density Filter
- **quadrature**: Unscented Kalman Filter [JU97], sigma-point filters [WM00; Mer04], Cubature and Quadrature Kalman filters [Ito00; AHE07]
- **Monte Carlo**: particle filter, Sequential Importance Sampling [HM54] and Resampling [GSS93], ensemble Kalman filter [Ell94]

Kalman Filter

The Kalman filter specifies both the state model and measurement model as linear, Gaussian.

$$\begin{tikzcd} Z_{t-1}|X_{1:t-1} \arrow[r, "\eta_d(z_t;Az_{t-1},\Gamma)"] & Z_t \arrow[d, "\eta_n(x_t;Hz_t,\Lambda)"] \\ & X_t \end{tikzcd}$$

If $Z_{t-1}|X_{1:t-1} \sim \mathcal{N}(v_{t-1}, \Phi_{t-1})$ then

$$Z_t|X_{1:t} \sim \mathcal{N}\left(\Phi_t(H^\top \Lambda^{-1} x_t + \hat{\Phi}_{t-1}^{-1} \hat{v}_{t-1}), \Phi_t\right).$$

Integration is accomplished with matrix multiplication.

N.B.: $\hat{v}_{t-1} = Av_{t-1}$, $\hat{\Phi}_{t-1} = A\Phi_{t-1}A^\top + \Gamma$, and $\Phi_t = (H^\top \Lambda^{-1} H + \hat{\Phi}_{t-1}^{-1})^{-1}$.

The Apollo Lunar Module used a variant of the Kalman Filter to land my fellow Purdue grad on the moon [Hal66; Hoa69; BL70].

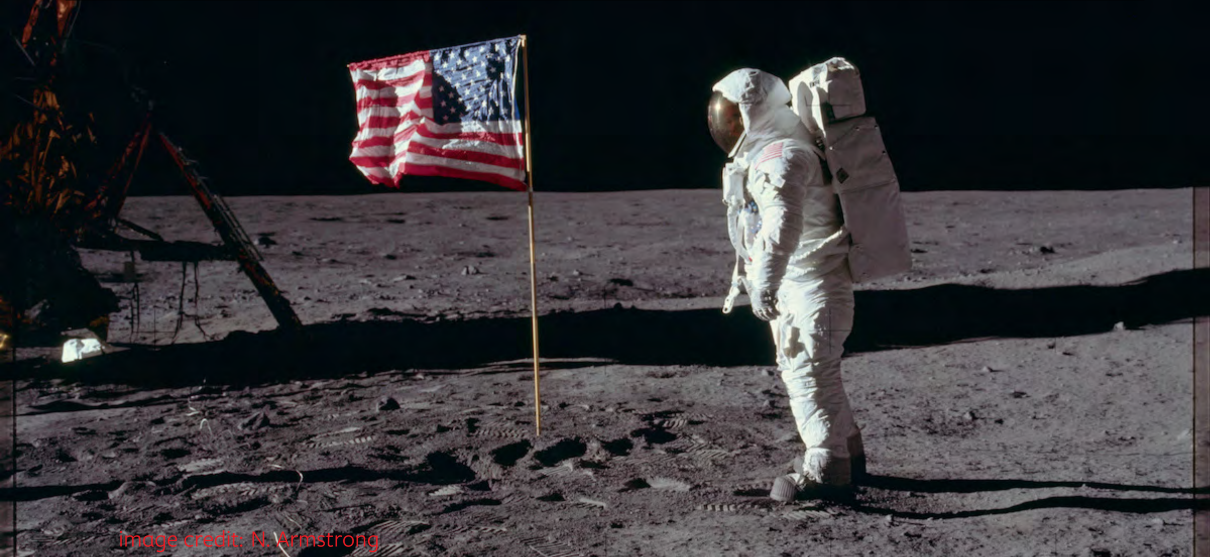
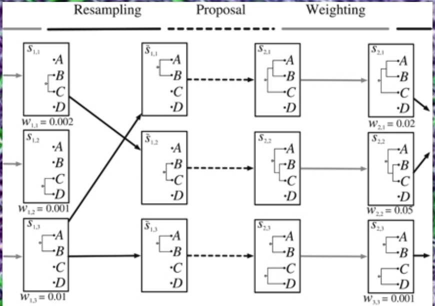


image credit: N. Armstrong

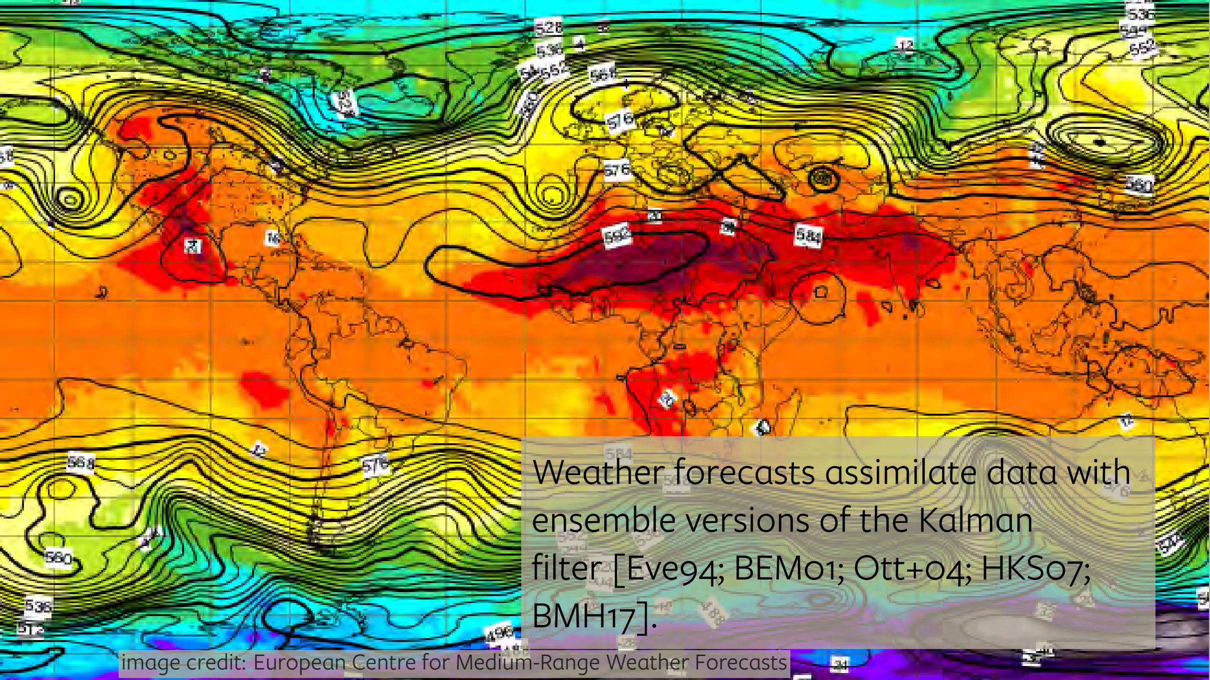
GPS receivers use the Extended Kalman filter to model and mitigate satellite clock offset and atmospheric delays [AB95; HLC01].

image credit: NASA



Sequential Monte Carlo is used to estimate phylogenetic trees [BSJ12; WBD15; DDM18].

image credit: NIH; phylogenetics diagram from [BSJ12]



Weather forecasts assimilate data with ensemble versions of the Kalman filter [Eve94; BEM01; Ott+04; HKS07; BMH17].

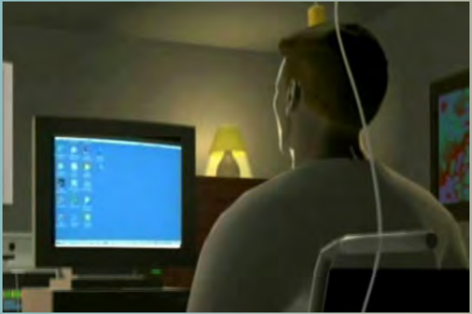
image credit: European Centre for Medium-Range Weather Forecasts

General Challenges to Filtering

- The approximations made by these filters can be quite poor.
  - The linear specification of the Kalman filter is very restrictive.
  - The EKF (that linearizes a nonlinear model) depends on its linearization.
  - Better approximations tend to require more computational time/resources.
- Implementing any of these filters *requires a generative probabilistic model of the process.*
  - A filter will only be as good as the model it is given.
  - Filtering approaches take the model as an input: they provide no guidance on how to learn that model.

Application: Neural Filtering

Filter updates need to be calculated in under 20ms.



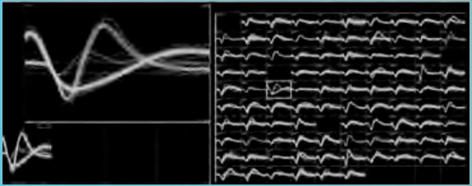
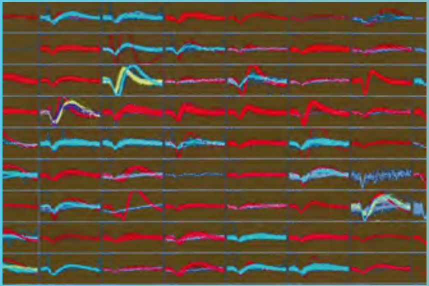


image credit: [Sun+05]

The relationship between observations and latent states can be nonlinear. Observations are very high-dimensional.

The Learning Problem

Generative approaches to filtering require a model for $p(x_t|z_t)$ ...



$p(x_t|z_t)$

$p(z_t|x_t)$

X_t, neural signals in $\mathbb{R}^{100}$

Z_t, intentions in $\mathbb{R}^3$

...but what if we could better model $p(z_t|x_t)$?

image credit: BrainGate

Discriminative Approach

We apply Bayes' rule:

$$p(z_t|x_{1:t}) \propto \underbrace{p(x_t|z_t)} \int p(z_t|z_{t-1})\; p(z_{t-1}|x_{1:t-1})\; dz_{t-1}$$

$$\propto \overbrace{\frac{p(z_t|x_t)}{p(z_t)}} \int p(z_t|z_{t-1})\; p(z_{t-1}|x_{1:t-1})\; dz_{t-1}$$

Discriminative Kalman Filter (DKF)

The DKF retains a linear, Gaussian state model, but takes $p(z_t|x_t) \propto \frac{p(z_t|x_t)}{p(z_t)}$ and approximates $p(z_t|x_t) \approx \eta_d(z_t; f(x_t), Q(x_t))$.

$$\begin{array}{ccc} Z_{t-1}|X_{1:t-1} & \xrightarrow{\eta_d(z_t;Az_{t-1},\Gamma)} & Z_t \\ & & \Big\downarrow \approx \frac{\eta_d(z_t;f(x_t),Q(x_t))}{\eta_d(z_t;0,S)} \\ & & X_t \end{array}$$

If $Z_{t-1}|X_{1:t-1} \sim \mathcal{N}(\mu_{t-1}, \Sigma_{t-1})$ then

$$Z_t|X_{1:t} \sim \mathcal{N}(\mu_t, \Sigma_t)$$

The DKF: a nonlinear relationship between Z_t and X_t with closed-form updates.

N.B.: $M_t = A\Sigma_{t-1}A^\intercal + \Gamma$, $\Sigma_t = (Q(x_t)^{-1} + M_t^{-1} - S^{-1})^{-1}$, and $\mu_t = \Sigma_t(Q(x_t)^{-1}f(x_t) + M_t^{-1}A\mu_{t-1})$ for $(Q(x_t)^{-1} - S^{-1})^{-1}$ pos.-definite.

Normality & the Bayesian CLT

For $\Theta \sim \text{Uniform}([0,1])$, draw a sequence:

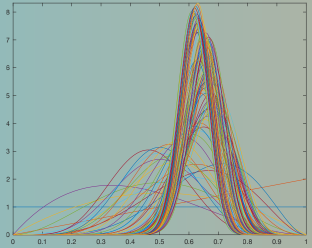
$$X_1, X_2 \ldots \sim^{\text{i.i.d.}} \text{Bernoulli}(\Theta)$$

Then:

$$p(\theta | x_1, \ldots, x_n) \propto \theta^{n\bar{x}}(1-\theta)^{n(1-\bar{x})}\mathbb{1}_{[0,1]}(\theta)$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ so that $\Theta | X_1, \ldots, X_n \sim \text{Beta}(n\bar{x}+1, n(1-\bar{x})+1)$. This distribution has mode $\bar{x}$ and variance bounded by $1/(n+2) \longrightarrow 0$.

I sampled $X_1, \ldots, X_{100} \sim^{\text{i.i.d.}} \text{Bernoulli}(0.6)$ and plotted the conditional densities $p(\theta | X_1 = x_1, \ldots, X_n = x_n)$ for $n = 1, \ldots, 100$ on the right. The bell shape is well-established by $n = 10$.



Bernstein–von Mises Theorem

We have reason to believe $p(z_t|x_t)$ is better-approximated as Gaussian than $p(x_t|z_t)$.

The number of observed dimensions is growing rapidly.

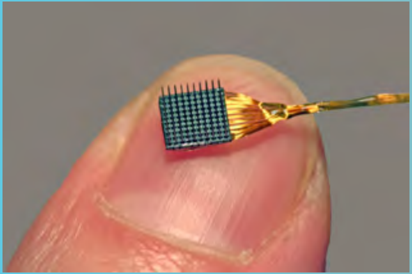


image credit: BrainGate

The Bernstein–von Mises Theorem (Bayesian CLT) provides mild conditions under which $p(z_t|x_t)$ becomes Gaussian in the limit as $\dim(X_t) \to \infty$ [Vaa98].

Theoretical Assurances

One potential concern might be that the renormalization step (dividing by a small quantity) could amplify approximation errors. We can show that this is not a problem.

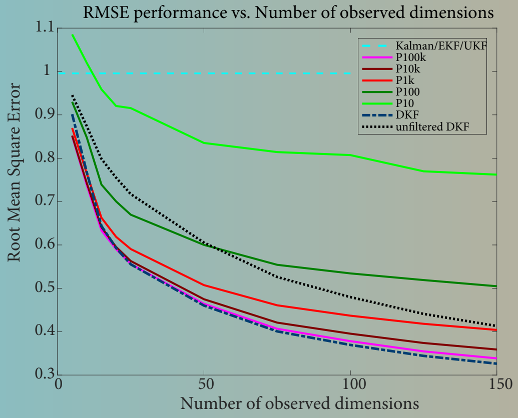
Result. *Under mild assumptions, the total variation between our approximation for* $p(z_t|x_{1:t})$ *and the true distribution converges in probability to zero as* $n \longrightarrow \infty$.

Model Validation: an Example

Consider a Kalman state model and a measurement model given by a mixture of linear Gaussians

$$p(x_t|z_t) = \sum_{\ell=1}^{L} \pi_\ell \eta_n(x_t; H_\ell z_t, \Lambda_\ell).$$

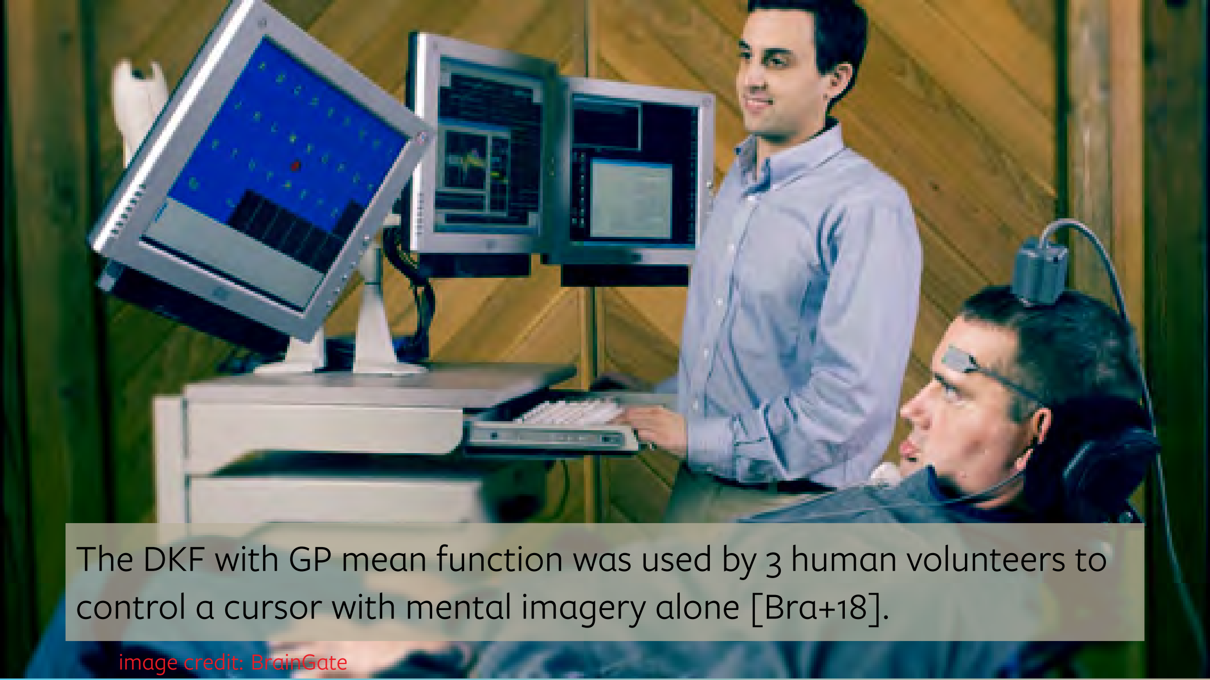
For parameters that make X_t and Z_t uncorrelated (but dependent), the Kalman/EKF/UKF filters are completely ineffective [Kus67].



Learning $p(z_t|x_t)$

Given a training set of (x_i, z_i) pairs, we experimented with various supervised methods to learn the functions $f(\cdot)$ and $Q(\cdot)$.

- **Nadaraya–Watson kernel regression**: a kernel-density estimation approach [Nad64; Wat64]
- **Neural network**: parametric and thus relatively better-suited to larger training sets [Bis06]
- **Gaussian process regression**: method ultimately implemented at BrainGate [RW06]



The DKF with GP mean function was used by 3 human volunteers to control a cursor with mental imagery alone [Bra+18].

image credit: BrainGate

Cross-country Neural Decoding

Nuyujukian et al. used the DKF-GP decoder to facilitate participant T9's use of a tablet computer for sending messages to another BrainGate participant based in California.

The Problem of Nonstationarities

The relationship between neural signals and user intention can change over the course of mere hours or minutes [Per+13; Per+14].

$$\begin{tikzcd}
Z_1 \arrow[r] \arrow[d] & Z_2 \arrow[r] \arrow[d] & \cdots \arrow[r] & Z_{n-1} \arrow[r] \arrow[d, squiggly] & Z_n \arrow[d, squiggly] \\
X_1 & X_2 & & X_{n-1} & X_n
\end{tikzcd}$$

used to learn $p(z_t|x_t)$

...but there was a change

Approach 1. Closed-loop decoder adaptation

- idea: continuously re-train the model with new data
- requires a nonstationary to be present for some time (during which filter performs poorly)
- can adapt to any arbitrary change in relationship

cf. [Jar+15; Bac+15; Gil+15; Ors+14; SLV08; Sha+14; SOC16; Dan+11]

Approach 2: Train a Robust Model

Alternatively:

- idea: train a model that is robust to certain types of nonstationarity
- automatically handles nonstationarities
- does not require online feedback or online retraining
- the amount change that can be overcome in this way is limited

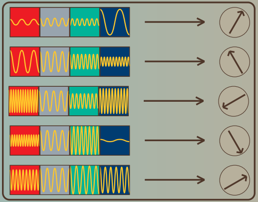
Sussillo et al. [Sus+12; Sus+16] piloted this approach by learning a stateful RNN for primate neural decoding (using data augmentation).

Kernel Selection for Robustness

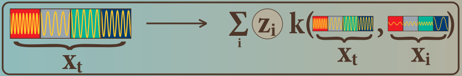
1. Given a supervised training set of (x_i, z_i) neural-intention pairs, we use a kernel regression estimator to obtain

$$\mathbb{E}[Z_t | X_t = x_t] \propto \sum_{i=1}^{n} z_i k(x_t, x_i).$$

2. The DKF then filters this estimate to obtain $\mathbb{E}[Z_t | X_{1:t} = x_{1:t}]$.



The above toy training bank of neural-intention pairs yields the following estimator for the test point x_t:

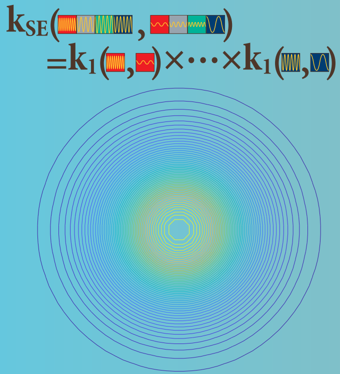


Kernel Selection

idea: to protect against single-neuron dropout/wonkiness, find a kernel that ignores large differences along a single dimension

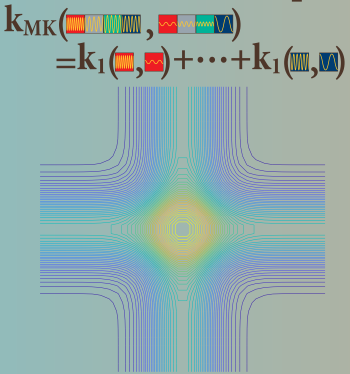
Standard Gaussian

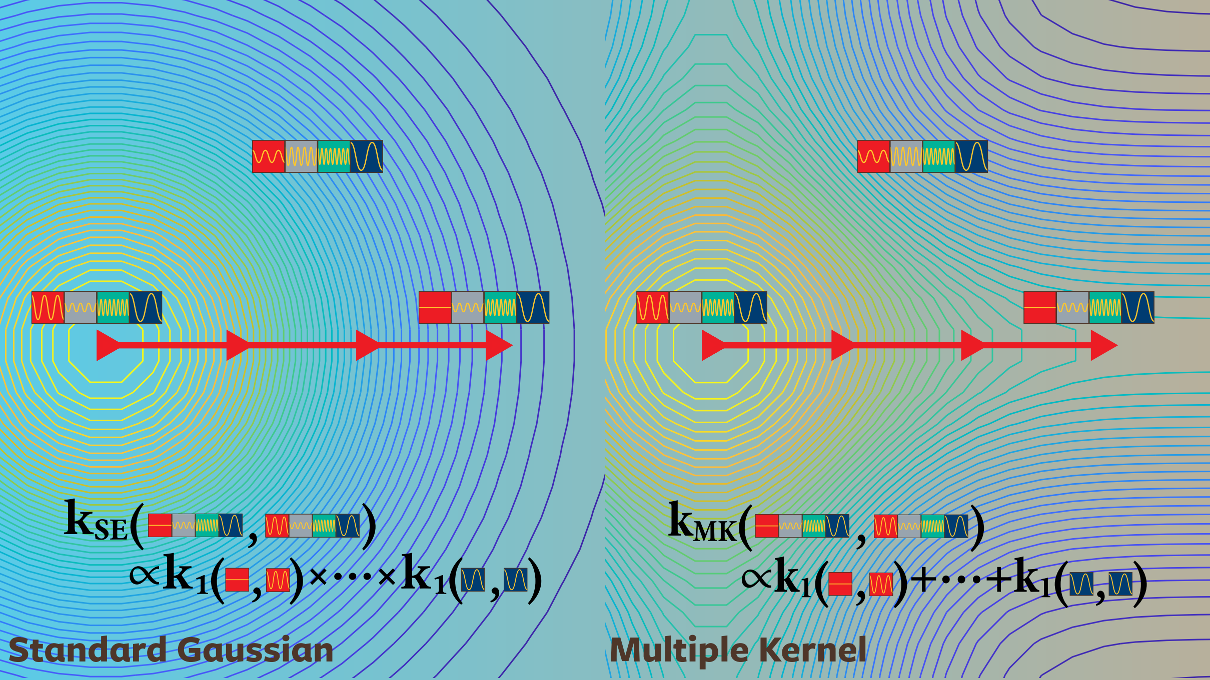
product of 1-d Gaussians



Multiple Kernel

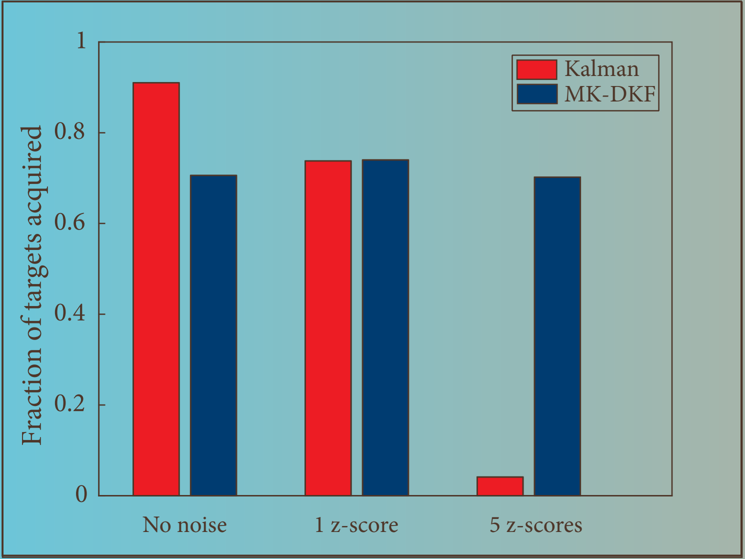
sum of 1-d Gaussians [GA11]





$k_{SE}($, $)$
$\propto k_1($, $) \times \cdots \times k_1($, $)$
$k_{MK}($, $)$
$\propto k_1($, $) + \cdots + k_1($, $)$
Standard Gaussian
Multiple Kernel

Results from Online Noise Injection Experiment with T10



Litany of the Saints

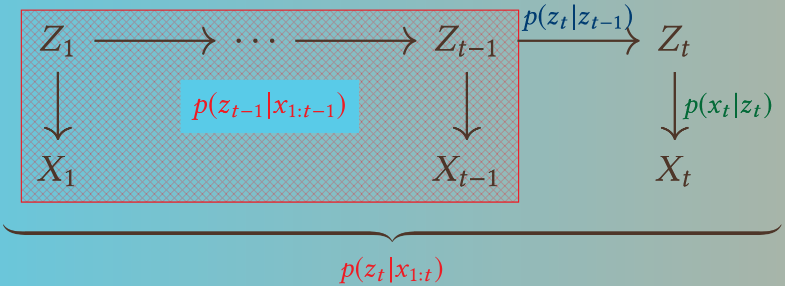
- ⚓ my parents
- ⚓ Matthew Harrison
- ⚓ Basilis Gidas
- ⚓ Jerome Darbon
- ⚓ Johnny Guzmán
- ⚓ Elizabeth Crites
- ⚓ Michael Snarski
- ⚓ Ian Alevy
- ⚓ Sameer Iyer
- ⚓ Cat Munro
- ⚓ Clark Bowman
- ⚓ Dan Keating
- ⚓ David Brandman
- ⚓ James Beatty
- ⚓ Chris Grimm
- ⚓ Leigh Hochberg
- ⚓ Kathleen Helbing
- ⚓ Robert Zink
- ⚓ Burgess Davis
- ⚓ Hans Uli Walther
- ⚓ Daniel Ocone
- ⚓ Eduardo Sontag
- ⚓ Victor Zavala
- ⚓ John Wiegand & family
- ⚓ Chris O'Neil & family
- ⚓ Ivana Petrovic
- ⚓ Nathan Meyers
- ⚓ Guo-Jhen Wu
- ⚓ Yuliya Mironovas
- ⚓ Beth Travers
- ⚓ Jean Radican
- ⚓ Stephanie Han
- ⚓ David Rosler
- ⚓ John Simeral
- ⚓ T9 & family
- ⚓ T10 & family
- ⚓ Jad Saab
- ⚓ Daniel Milstein
- ⚓ Ed Chien
- ⚓ Isaac Solomon
- ⚓ Nobuyuki Kaya
- ⚓ Brittany Sorice
- ⚓ Jessica Kelemen
- ⚓ Brian Franco
- ⚓ Beata Jarosiewicz
- ⚓ Carlos Vargas-Irwin
- ⚓ Arthur Gretton
- ⚓ Tommy Hosman
- ⚓ Benjamin Shanahan

And so many others!

This thesis is dedicated to BrainGate participant T9.

Thanks for joining us today!

Bayesian Filtering Recursion



Kalman Filter Derivation

If the measurement and observation models are linear, Gaussian

$$p(z_t|z_{t-1}) = \eta_d(z_t; Az_{t-1}, \Gamma) \quad \text{and} \quad p(x_t|z_t) = \eta_n(x_t; Hz_t, \Lambda)$$

then (if z_t stationary), the posterior $p(z_t|x_{1:t}) = \eta_d(z_t; v_t, \Phi_t)$ is also Gaussian and we have:

$$p(z_t|x_{1:t}) \propto p(x_t|z_t) \int p(z_t|z_{t-1})\, p(z_{t-1}|x_{1:t-1})\, dz_{t-1}$$

$$\eta_d(z_t; v_t, \Phi_t) \propto \eta_n(x_t; Hz_t, \Lambda) \underbrace{\int \eta_d(z_t; Az_{t-1}, \Gamma)\, \eta_d(z_{t-1}; v_{t-1}, \Phi_{t-1})\, dz_{t-1}}_{p(z_t|x_{1:t-1}) \propto \eta_d(z_t; Av_{t-1}, A\Phi_{t-1}A^\intercal + \Gamma)}$$

$$\propto \eta_d\left(z_t; \Phi_t(H^\intercal \Lambda^{-1} x_t + \hat{\Phi}_{t-1}^{-1} \hat{v}_{t-1}), \Phi_t\right)$$

where $\hat{v}_{t-1} = Av_{t-1}$, $\hat{\Phi}_{t-1} = A\Phi_{t-1}A^\intercal + \Gamma$, and

$$\Phi_t = (H^\intercal \Lambda^{-1} H + \hat{\Phi}_{t-1}^{-1})^{-1}$$

see Kalman [Kal60] and Kalman and Bucy [KB61] for the original papers

Kalman II

By defining the Kalman gain as

$$K_t := \hat{\Phi}_{t-1} H^\top (H \hat{\Phi}_{t-1} H^\top + \Lambda)^{-1}$$

it follows that

$$\Phi_t = (I_d - K_t H) \hat{\Phi}_{t-1}$$

and

$$v_t = \hat{v}_{t-1} + K_t (x_t - H \hat{v}_{t-1}).$$

Kalman III

Note that the Kalman model implies

$$p(x_t|x_{1:t-1}) = \eta_n(x_t; H\hat{v}_{t-1}, H\hat{\Phi}_{t-1}H^\intercal + \Lambda).$$

Let $\bar{X}_t, \bar{Z}_t$ be distributed as X_t, Z_t conditioned on $X_{1:t-1}$, respectively. Then

$$\mathbb{V}[\bar{X}_t] = H\hat{\Phi}_{t-1}H^\intercal + \Lambda,$$
$$\mathrm{Cov}[\bar{Z}_t, \bar{X}_t] = \hat{\Phi}_{t-1}H^\intercal,$$

so that we can re-express the Kalman gain, posterior covariance, and posterior mean as:

$$\begin{aligned} K_t &= \mathrm{Cov}[\bar{Z}_t, \bar{X}_t](\mathbb{V}[\bar{X}_t])^{-1}, \\ \Phi_t &= \mathbb{V}[\bar{Z}_t] - K_t\, \mathbb{V}[\bar{X}_t]K_t^\intercal, \\ v_t &= \mathbb{E}[\bar{Z}_t] + K_t(x_t - \mathbb{E}[\bar{X}_t]). \end{aligned}$$

This will form the basis for the Gaussian assumed density filter (and UKF).

Extended Kalman Filter

If the measurement and observation models remain Gaussian, but are now allowed to be nonlinear:

$$p(z_t|z_{t-1}) = \eta_d(z_t; a(z_{t-1}), \Gamma) \quad \text{and} \quad p(x_t|z_t) = \eta_n(x_t; h(z_t), \Lambda)$$

Then we can, at every time step, we can form a linear approximation at the prior mean and use that instead, i.e.:

$$p(z_t|z_{t-1}) \approx \eta_d(z_t; a(v_{t-1}) + \tilde{A}(z_t - v_{t-1}), \Gamma) \text{ and } p(x_t|z_t) = \eta_n(x_t; h(\hat{v}_{t-1}) + \tilde{H}(z_t - \hat{v}_{t-1}), \Lambda)$$

where $\tilde{A}$ and $\tilde{H}$ are the Jacobians of a, h evaluated at the respective prior means.

see Gelb [Gel74] for details

Laplace Approximation

This Taylor series expansion in the exponent replaces a pdf with a Gaussian density that matches curvature at the mode: At step t,

$$p(z_t|x_{1:t}) \propto p(x_t|z_t) \int p(z_t|z_{t-1})\, p(z_{t-1}|x_{1:t-1})\, dz_{t-1} =: r(z_t)$$

where $p(z_t|x_{1:t-1}) = \eta(z_t; \nu, \tau^2)$ so that

$$g(z_t) := \log(r(z_t)) = -\frac{1}{2\tau^2}(z_t - \nu)^2 + \log(p(x_t|z_t))$$

where $r(z_t) = e^{g(z_t)}$. We find $z_t^* = \arg\max_z g(z)$ and form the Laplace approximation

$$r_1(z_t) \approx e^{g(z_t^*)+g'(z_t^*)(z-z_t^*)+g''(z_t^*)(z-z_t^*)^2/2} = \eta(z_t; z_t^*, -1/g''(z_t^*))$$

where $g'(z_t^*) = 0$ because z_t^* is an extremal point and $g''(z_t^*) < 0$ because z_t^* is a maximum.

see Butler [But07] for details

Gaussian Assumed Density Filter

Instead of solving for the posterior, we can find a Gaussian that most closely approximates it in the sense of KL-divergence:

$$v_t, \Phi_t = \arg\min_{a,b} \left\{ D_{KL}\big(p(z_t | x_{1:t}) || \eta_d(z_t; a, b)\big) \right\}$$

This approach yields:

$$\begin{aligned} K &= P_{zx} P_{xx}^{-1}, \\ \Phi_t &= \hat{\Phi}_{t-1} - K P_{xx} K^\top, \\ v_t &= \hat{v}_{t-1} + K(x_t - \mu_x). \end{aligned}$$

where:

$$\begin{aligned} \mu_x &= \int h(z_t)\, \eta_d(z_t; \hat{v}_{t-1}, \hat{\Phi}_{t-1})\, dz_t, \\ P_{xx} &= \int (h(z_t) - \mu_x)(h(z_t) - \mu_x)^\top\, \eta_d(z_t; \hat{v}_{t-1}, \hat{\Phi}_{t-1})\, dz_t + \Lambda, \\ P_{zx} &= \int (z_t - \hat{v}_{t-1})(h(z_t) - \mu_x)^\top\, \eta_d(z_t; \hat{v}_{t-1}, \hat{\Phi}_{t-1})\, dz_t. \end{aligned}$$

for $Z_t \mid X_{1:t-1} \sim \mathcal{N}(\hat{v}_{t-1}, \hat{\Phi}_{t-1})$.

see, e.g., Kushner [Kus67]

Gaussian Assumed Density Filter II

In other words, we have:

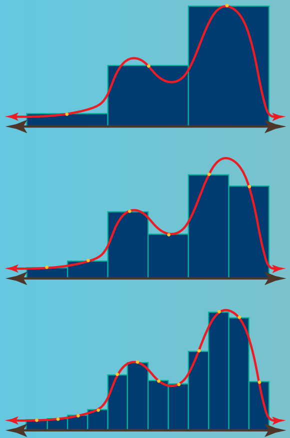
$$
\begin{aligned}
K &= \mathrm{cov}[\hat{Z}_t, h(\hat{Z}_t)](\mathbb{V}[h(\hat{Z}_t)] + \Lambda)^{-1}, \\
\Phi_t &= \hat{\Phi}_{t-1} - K(\mathbb{V}[h(\hat{Z}_t)] + \Lambda)K^\intercal, \\
v_t &= \hat{v}_{t-1} + K(x_t - \mathbb{E}[h(\hat{Z}_t)]).
\end{aligned}
$$

where $\hat{Z}_t \sim \mathcal{N}(\hat{v}_{t-1}, \hat{\Phi}_{t-1})$. Nonlinear state updates are handled as follows:

$$
\begin{aligned}
\hat{v}_{t-1} &= \mathbb{E}[a(\bar{Z}_t)]) \\
\hat{\Phi}_{t-1} &= \mathbb{V}[a(\bar{Z}_t)])
\end{aligned}
$$

where $\bar{Z}_t \sim \mathcal{N}(v_{t-1}, \Phi_{t-1})$.

Quadrature



see [CBK00; IX00; AHE07; AH09]

- quadrature converts integrals to sums by evaluating the integrand at a deterministic set of points (anchors)
- for example, the midpoint quadrature rule estimates:

$$\int_a^b f(x)\,dx \approx (b-a)\cdot f\left(\tfrac{a+b}{2}\right)$$

- the Gaussian quadrature rule uses a clever choice of weights and points to make this approximation exact for all polynomials up to a certain degree [Gol73]
- many quadrature rules have been used to calculate integrals for the Gaussian ADF

Unscented Kalman Filter

The UKF propagates $2d + 1$ weighted points through h to estimate the integrals in the Gaussian assumed density filter, in a method known as the unscented transform.

We introduce parameters $\alpha > 0, \beta \in \mathbb{R}$ and consider the set of sigma vectors $\zeta_0, \ldots, \zeta_{2d} \in \mathbb{R}^d$ given by

$$\zeta_0 = \hat{v}_{t-1},$$

$$\zeta_i = \hat{v}_{t-1} + \left(\sqrt{\alpha^2 d \hat{\Phi}}\right)_i, \qquad i = 1, \ldots, d$$

$$\zeta_i = \hat{v}_{t-1} - \left(\sqrt{\alpha^2 d \hat{\Phi}}\right)_i, \qquad i = d+1, \ldots, 2d$$

where $\left(\sqrt{\alpha^2 d \hat{\Phi}}\right)_i$ denotes the ith row of the matrix square root.

We set weights $w_0^{(m)} = 1 - 1/\alpha^2$, $w_0^{(c)} = 2 - 1/\alpha^2 - \alpha^2 + \beta$, and $w_0^{(m)} = w_0^{(c)} = 1/(2\alpha^2 d)$. Then:

$$\mu_x = \sum_{i=0}^{2d} w_i^{(m)} h(\zeta_i),$$

$$P_{xx} = \sum_{i=0}^{2d} w_i^{(c)} (h(\zeta_i) - \bar{x})(h(\zeta_i) - \bar{x})^\intercal,$$

$$P_{zx} = \sum_{i=0}^{2d} w_i^{(c)} (\zeta_i - \hat{v}_{t-1})(h(\zeta_i) - \bar{x})^\intercal.$$

see Julier and Uhlmann [JU97] for original paper; Wan and Merwe [WM00] for details

Sequential Importance Sampling

Representing $p(z_{t-1}|x_{1:t-1}) \approx \sum_{\ell=1}^{L} w_{t-1}^{(\ell)} \delta_{z_{t-1}^{(\ell)}}(z_{t-1})$ as the weighted sum of particles, we have:

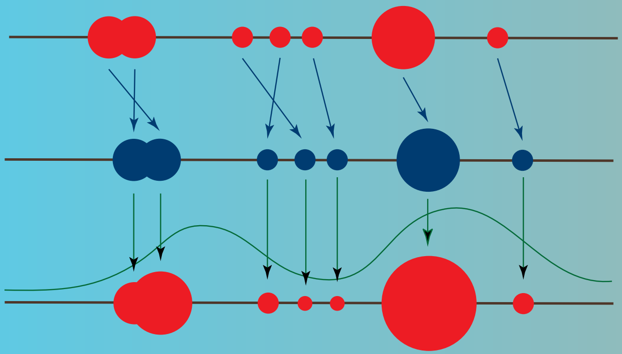
$$p(z_t|x_{1:t}) \propto p(x_t|z_t) \int p(z_t|z_{t-1}) \sum_{\ell=1}^{L} w_{t-1}^{(\ell)} \delta_{z_{t-1}^{(\ell)}}(z_{t-1})\, dz_{t-1}$$

$$\propto p(x_t|z_t) \sum_{\ell=1}^{L} w_{t-1}^{(\ell)} \delta_{z_t^{(\ell)}}(z_t), \quad \text{where } z_t^{(\ell)} \sim Z_t|\{Z_{t-1} = z_{t-1}^{(\ell)}\}$$

$$\propto \sum_{\ell=1}^{L} w_t^{(\ell)} \delta_{\hat{z}_{t-1}^{(\ell)}}(z_t), \quad \text{where } w_t^{(\ell)} \propto w_{t-1}^{(\ell)} \cdot p(X_t = x_t|Z_t = z_t^{(\ell)}).$$

see Moral [Mor96] and Doucet, Godsill, and Andrieu [DGA00] for details

Sequential Importance Sampling II



prior at time $t - 1$

sample from state model

re-weight according to
measurement model

Sequential Importance Resampling

In SIS, most of the weight can be placed a very few number of particles, leading to a problem called weight degeneracy. A resampling step was introduced to fix that problem.

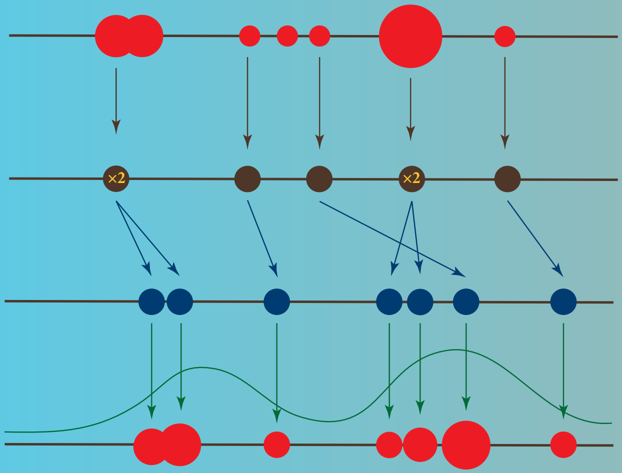
$$
\begin{aligned}
p(z_t|x_{1:t}) &\propto p(x_t|z_t) \int p(z_t|z_{t-1}) \sum_{\ell=1}^{L} w_{t-1}^{(\ell)} \delta_{z_{t-1}^{(\ell)}}(z_{t-1})\, dz_{t-1} \\
&\propto p(x_t|z_t) \int p(z_t|z_{t-1}) \sum_{\ell=1}^{L} \delta_{\tilde{z}_{t-1}^{(\ell)}}(z_{t-1})\, dz_{t-1},
\end{aligned}
$$

where now $\tilde{z}_{t-1}^{(\ell)}$ are drawn, with replacement, from the $z_{t-1}^{(\ell)}$ with relative odds $w_{t-1}^{(\ell)}$,

$$
\begin{aligned}
&\propto p(x_t|z_t) \sum_{\ell=1}^{L} \delta_{z_t^{(\ell)}}(z_t), \qquad \text{where } z_t^{(\ell)} \sim Z_t|\{Z_{t-1} = \tilde{z}_{t-1}^{(\ell)}\} \\
&\propto \sum_{\ell=1}^{L} w_t^{(\ell)} \delta_{\hat{z}_{t-1}^{(\ell)}}(z_t), \qquad \text{where } w_t^{(\ell)} \propto p(X_t = x_t|Z_t = z_t^{(\ell)}).
\end{aligned}
$$

see Gordon, Salmond, and Smith [GSS93]

Sequential Importance Resampling II



prior at time $t-1$

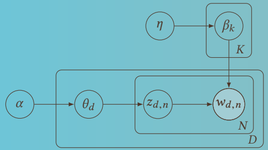
resample, reset weights

sample from state model

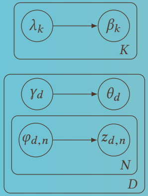
re-weight according to measurement model

Variational Inference: an Illustrative Example

0. Given a generative model $p(\beta, \theta, z)$ that's intractable to integrate:



1. We can specify a (variational) model $q(\beta, \theta, z)$ that's easier to integrate (more independence):



2. We perform inference by tweaking the variational parameters to minimize KL divergence between the variational and true models:

$$\hat{\lambda}, \hat{\gamma}, \hat{\varphi} = \arg\min_{\lambda,\gamma,\varphi} \{D\left(q(\beta, \theta, z) \| \, p(\beta, \theta, z)\right)\}$$

3. Now, inference has become an optimization problem instead of an integration problem.

The Discriminative Kalman Filter

Under a stationary Kalman state model

$$\begin{aligned} p(z_0) &= \eta_d(z_t; \vec{0}, S) \\ p(z_t|z_{t-1}) &= \eta_d(z_t; Az_{t-1}, \Gamma) \end{aligned}$$

and a discriminative observation model

$$p(z_t|x_t) \approx \eta_d(z_t; f(x_t), Q(x_t))$$

the posterior will also be Gaussian.

At every time step t, we have

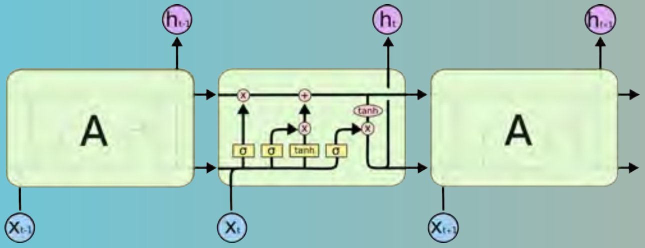
$$\begin{aligned} p(z_t|x_{1:t}) &\approx \eta_d(z_t; \mu_t(x_{1:t}), \Sigma_t(x_{1:t})) \\ &\approx \eta_d(z_t; \mu_t, \Sigma_t) \end{aligned}$$

related sequentially via the closed-form updates

$$\begin{aligned} M_t &= A\Sigma_{t-1}A^\intercal + \Gamma, \\ \Sigma_t &= (Q(x_t)^{-1} + M_t^{-1} - S^{-1})^{-1}, \\ \mu_t &= \Sigma_t(Q(x_t)^{-1} f(x_t) + M_t^{-1} A\mu_{t-1}) \end{aligned}$$

if $(Q(x_t)^{-1} - S^{-1})^{-1}$ is pos.-definite.

Nonprobabilistic Filtering: the Long Short-Term Memory (LSTM) Recursive Neural Network



- designed these to overcome error backflow problems [HS97]
- dropout [Sri+14] should only applied to feedforward connections, not recurrent connections [Pha+14; ZSV14]
- Batch-normalization to prevent covariate shift [IS15]
- Xavier-type parameter initialization [GB10]
- Adadelta [Zei12] was designed to improve Adagrad [DHS11] by allowing learning rate to sometimes increase

image credit: Olah [Ola15]

LSTM implementation in TensorFlow: highlights

```python
def lstm_step(inp, prev, state):
    with tf.name_scope('dimensionality'):
        dstate = state.get_shape()[1].__int__()
        din = inp.get_shape()[1].__int__()
        dout = prev.get_shape()[1].__int__()
        gates = {}
    for g in ['forget', 'input', 'output', 'state']:
        with tf.name_scope(g):
            W = tf.Variable(tf.truncated_normal([din, dstate], stddev=1 / tf.sqrt(tf.to_float(din))))
            U = tf.Variable(tf.truncated_normal([dout, dstate], stddev=1 / tf.sqrt(tf.to_float(dout))))
            b = tf.Variable(tf.zeros([1, dstate]))
            combo = tf.matmul(inp, W) + tf.matmul(prev, U) + b
            if g in ['forget', 'input', 'output']:
                gates[g] = tf.sigmoid(combo)
            else:
                state = tf.multiply(gates['forget'], state) + tf.multiply(gates['input'], tf.tanh(combo))
    with tf.name_scope('output'):
        W = tf.Variable(tf.truncated_normal([dstate, dout], stddev=1 / tf.sqrt(tf.to_float(dstate))))
        b = tf.Variable(tf.zeros([1, dout]))
        outp = tf.matmul(tf.multiply(gates['output'], tf.tanh(state)), W) + b
    return outp, state


def lstm_full(inp, dout, dstate):
    with tf.name_scope('dimensionality'):
        T = len(inp)
        n = inp[0].get_shape()[0]
    with tf.name_scope('states'):
        state = tf.Variable(tf.zeros([n, dstate]), trainable=False)
        out = tf.Variable(tf.zeros([n, dout]), trainable=False)
    for t in range(T):
        with tf.name_scope('step' + str(t)):
            out, state = lstm_step(inp[t], out, state)
            state = tf.layers.batch_normalization(state)
            out = tf.layers.batch_normalization(out)
            state = tf.nn.dropout(state, keep_prob=keep_prob_)
            out = tf.nn.dropout(out, keep_prob=keep_prob_)
    return out
```

Nadaraya–Watson Kernel Regression

Given a dataset $\{(x_i, z_i)\}_{i=1}^n$, we can use Kernel Density Estimation (KDE) to model the joint density:

$$p(z, x) \approx \frac{1}{n} \sum_{i=1}^{n} \kappa_Z(z, z_i)\kappa_X(x, x_i).$$

The conditional distribution is then

$$\begin{aligned} p(z|x) &= \frac{p(z, x)}{p(x)} \\ &\approx \frac{\sum_{i=1}^m \kappa_Z(z, Z_i)\kappa_X(x, X_i)}{\sum_{i=1}^m \kappa_X(x, X_i)}. \end{aligned}$$

It follows that the conditional mean is

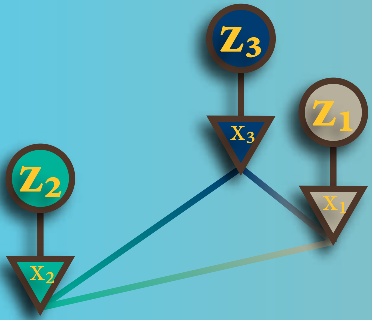
$$\begin{aligned} \mathbb{E}[Z|X = x] &= \int zp(z|x)dz \\ &\approx \frac{\sum_{i=1}^m \left(\int z\kappa_Z(z, Z_i)dz\right)\kappa_X(x, X_i)}{\sum_{i=1}^m \kappa_X(x, X_i)} \\ &\approx \frac{\sum_{i=1}^m Z_i\kappa_X(x, X_i)}{\sum_{i=1}^m \kappa_X(x, X_i)} \end{aligned}$$

for a symmetric kernel [Nad64; Wat64]. If the kernel is also stationary

$$\mathbb{E}[Z^\top Z|X = x] \approx \Sigma_Z + \frac{\sum_{i=1}^m Z_i^\top Z_i\kappa_X(x, X_i)}{\sum_{i=1}^m \kappa_X(x, X_i)}$$

from which we estimate the conditional variance.

Gaussian Process Model



We specify a covariance structure:

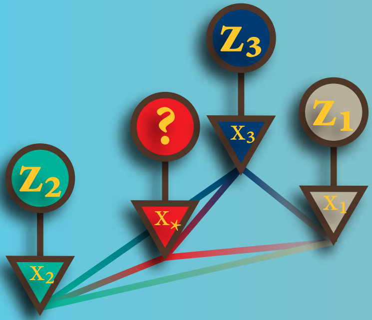
$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} + \sigma^2 I \right)$$

where

$$k_{ij} = k_\theta(x_i, x_j)$$

so that closer x-values correspond to more correlated z-values.

Gaussian Process Inference

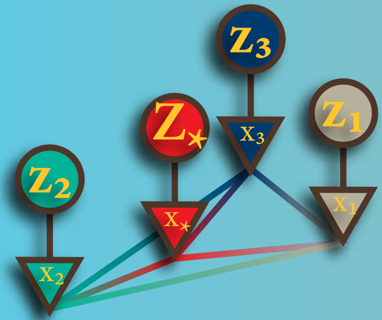


Given some test value x_*, we now have:

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ ? \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \tilde{k}_{11} & k_{12} & k_{13} & k_{1*} \\ k_{21} & \tilde{k}_{22} & k_{23} & k_{2*} \\ k_{31} & k_{32} & \tilde{k}_{33} & k_{3*} \\ k_{*1} & k_{*2} & k_{*3} & k_{**} \end{bmatrix} \right)$$

where $\tilde{k}_{ii} = k_{ii} + \sigma^2$ to account for noisy observations.

Gaussian Process Inference II



Given some test value x_*, we now have:

$$\begin{bmatrix} z \\ Z_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K + \sigma^2 I & k_* \\ k_*^\top & k_{**} \end{bmatrix} \right)$$

It follows that:

$$Z_* | \{X_* = x_*, \mathcal{D}\} \sim \mathcal{N}(k_*^\top (K + \sigma^2 I)^{-1} z, k_{**} - k_*^\top (K + \sigma^2 I)^{-1} k_*)$$

Bernstein–von Mises Theorem: Formal Statement

Let the experiment $(P_\theta : \theta \in \Theta)$ be differentiable in quadratic mean at θ_0 with nonsingular Fisher information matrix I_{θ_0}, and suppose that for every $\epsilon > 0$ there exists a sequence of tests ϕ_n such that:

$$P^n_{\theta_0}\phi_n \longrightarrow 0, \qquad \sup_{\|\theta-\theta_0\|\geq\epsilon} P^n_\theta(1-\phi_n) \longrightarrow 0$$

Furthermore, let the prior measure be absolutely continuous in a neighborhood of θ_0 with a continuous positive density at θ_0. Then the corresponding posterior distributions satisfy

$$\left\| P_{\sqrt{n}(\bar{\Theta}_n-\theta_0)|X_1,\ldots,X_n} - \mathcal{N}(\Delta_{n,\theta_0}, I^{-1}_{\theta_0}) \right\| \xrightarrow{P^n_{\theta_0}} 0$$

where $\Delta_{n,\theta_0} = \frac{1}{\sqrt{n}}\sum_{i=1}^n I^{-1}_{\theta_0}\dot{\ell}_{\theta_0}(X_i)$, $\dot{\ell}_{\theta_0}$ is the score function of the model, and the norm is that of total variation.

see [Vaa98]

Bernstein–von Mises Theorem: A Rephrasing

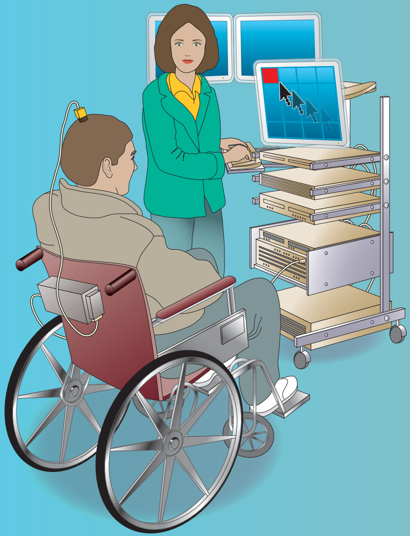
Total variation is invariant to location and scale changes, so that

$$\left\| P_{\bar{\Theta}|X_1,\ldots,X_n} - \mathcal{N}(\hat{\theta}_n, \tfrac{1}{n} I_\theta^{-1}) \right\| \xrightarrow{P_\theta^n} 0$$

For a random variable Θ, if we are given a sequence $X_1, X_2, \ldots$ of random variables that each tell us a little bit more about Θ, then $\bar{\Theta}|X_1, \ldots, X_n$ converges in a very strong sense to a Gaussian with mean $\hat{\theta}_n$.

see [Vaa98]

BrainGate setup



- BG uses the NeuroPort System (Blackrock Microsystems, Salt Lake City, UT) to sample raw neural signals for each electrode (30kHz)
- an xPC target real-time operating system (Mathworks, Natick, MA) then processes these signals
- signals are downsampled to 15kHz & de-noised by subtracting an instantaneous common average reference using 40 of the 96 channels on each array with the lowest RMS value
- de-noised signals were band-pass filtered between 250 Hz and 5000 Hz using an 8th order non-causal Butterworth filter
- features are binned in 20ms non-overlapping increments

image credit: BrainGate; for more details, see dissertation section on "Signal acquisition"

Challenges to Effective Decoding

- good offline filters do not necessarily make good closed-loop filters [Jar+13]
- metrics for BCI performance are non-standardized / lack of ground truth [Bil+13; Tho+14]
- the relationship between neural signals and intention changes over time [Per+13; Per+14]

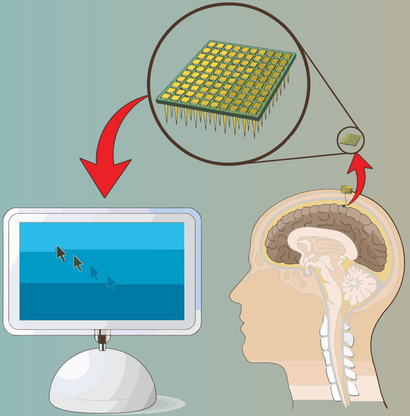
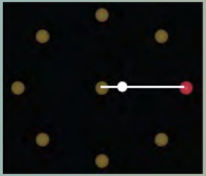


image credit: BrainGate

BrainGate Calibration

1. In the first 2-3 minutes, the computer moves the cursor and the participant imagines directing it ("open–loop").
2. In the next ~ 10 minutes, the participant controls the cursor with assistance (error attenuation) that helps guide the cursor to the target ("closed–loop with error attenuation"). With sequential retraining, the quality of data improves and the need for assistance decreases.
3. Finally, the assistance is turned off completely and the participant assumes full control over the cursor.



1. open–loop calibration

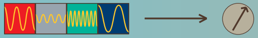
2. error attenuation

DKF implementation in Matlab: highlights

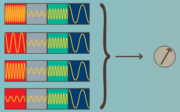
```
function means_filtered = DKF_filtering(means, vars, A, G, VZ)
%DKF_FILTERING filters under the DKF model:
% hidden(t)|observed(t) ~ N(means, vars)
% hidden(t)|hidden(t-1) ~ N(A*hidden(t-1), G)
% hidden(t) ~ N(0, VZ)
% this function returns the estimates:
% hidden(t)|observed(1:t-1) ~ N(means_filtered, vars_filtered)
% means are [n,T] dimensional
% vars are [n,n,T] dimensional

[n,T] = size(means);
means_filtered = zeros([n,T]);

S = vars(:,:,1);
nu = means(:,1);
means_filtered(:,1) = nu;

for t = 2:T
    aa = means(:,t);
    bb = vars(:,:,t);
    if ~all(eig(inv(bb)-inv(VZ))>0)
        bb = inv(inv(bb)+inv(VZ));
    end
    Mi = pinv(G+A*S*A');
    S = pinv(pinv(bb)+Mi-pinv(VZ));
    nu = S*(bb\aa+Mi*(A*nu));
    means_filtered(:,t) = nu;
end

end
```

Data Augmentation for Robustness

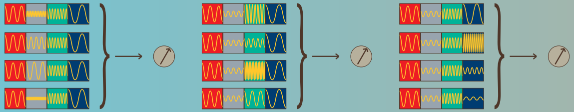
1. Take a datapoint:



2. Add noisy examples for neuron 1, with same label:



3. Repeat this process for all other neurons:



4. Add all these new examples to your training set

References I

- I. Arasaratnam and S. Haykin. "Cubature Kalman Filters." In: *IEEE Trans. Autom. Control* 54.6 (2009), pp. 1254–1269.
- I. Arasaratnam, S. Haykin, and R. J. Elliott. "Discrete-Time Nonlinear Filtering Algorithms Using Gauss–Hermite Quadrature." In: *Proc. IEEE* 95.5 (2007), pp. 953–977.
- P. Axelrad and R. G. Brown. "Global Positioning System: Theory and Applications." In: American Institute of Aeronautics and Astronautics, 1995. Chap. 9. GPS Navigation Algorithms.
- D. Bacher, B. Jarosiewicz, N. Y. Masse, S. D. Stavisky, J. D. Simeral, K. Newell, E. M. Oakley, S. S. Cash, G. Friehs, and L. R. Hochberg. "Neural Point-and-Click Communication by a Person With Incomplete Locked-In Syndrome." In: *Neurorehabil. Neural Repair* 29.5 (2015), pp. 462–471.
- R. H. Battin and G. M. Levine. "Application of Kalman filtering techniques to the Apollo program." In: *Theory and Applications of Kalman Filtering*. Ed. by C. T. Leondes. NATO, Advisory Group for Aerospace Research and Development, 1970. Chap. 14.
- V. E. Beneš. "Exact finite-dimensional filters for certain diffusions with nonlinear drift." In: *Stochastics* 5.1-2 (1981), pp. 65–92.
- M. Billinger, I. Daly, V. Kaiser, J. Jin, B. Allison, and G. Mueller-Putz. "Towards Practical Brain-Computer Interfaces." In: Springer, 2013. Chap. Is it significant? Guidelines for reporting BCI performance.
- C. H. Bishop, B. J. Etherton, and S. J. Majumdar. "Adaptive Sampling with the Ensemble Transform Kalman Filter. Part I: Theoretical Aspects." In: *Mon. Weather Rev.* 129.3 (2001), pp. 420–436.
- C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

References II

- A. Bouchard-Côté, S. Sankararaman, and M. I. Jordan. "Phylogenetic inference via sequential Monte Carlo." In: *Syst. Biol.* 61.4 (2012), pp. 579–593.
- D. M. Brandman, T. Hosman, J. Saab, M. C. Burkhart, B. E. Shanahan, J. G. Ciancibello, A. A. Sarma, D. J. Milstein, C. E. Vargas-Irwin, B. Franco, J. Kelemen, C. Blabe, B. A. Murphy, D. R. Young, F. R. Willett, C. Pandarinath, S. D. Stavisky, R. F. Kirsch, B. L. Walter, A. Bolu Ajiboye, S. S. Cash, E. E. Eskandar, J. P. Miller, J. A. Sweet, K. V. Shenoy, J. M. Henderson, B. Jarosiewicz, M. T. Harrison, J. D. Simeral, and L. R. Hochberg. "Rapid calibration of an intracortical brain–computer interface for people with tetraplegia." In: *J. Neural Eng.* 15.2 (2018).
- M. Buehner, R. McTaggart-Cowan, and S. Heilliette. "An Ensemble Kalman Filter for Numerical Weather Prediction Based on Variational Data Assimilation: VarEnKF." In: *Mon. Weather Rev.* 145.2 (2017), pp. 617–635.
- R. W. Butler. *Saddlepoint approximations with applications*. Vol. 22. Cambridge University Press, 2007.
- S. Challa, Y. Bar-Shalom, and V. Krishnamurthy. "Nonlinear filtering via generalized Edgeworth series and Gauss-Hermite quadrature." In: *IEEE Trans. Signal Process.* 48.6 (2000), pp. 1816–1820.
- S. Dangi, S. Gowda, R. Heliot, and J. M. Carmena. "Adaptive Kalman filtering for closed-loop Brain-Machine Interface systems." In: *Int. IEEE/EMBS Conf. Neural Eng.* (2011), pp. 609–612.
- F. E. Daum. "Exact finite dimensional nonlinear filters for continuous time processes with discrete time measurements." In: *IEEE Conf. Decis. Control*. 1984, pp. 16–22.
- F. E. Daum. "Exact finite-dimensional nonlinear filters." In: *IEEE Trans. Autom. Control* 31.7 (1986), pp. 616–622.
- V. Dinh, A. E. Darling, and F. A. Matsen IV. "Online Bayesian Phylogenetic Inference: Theoretical Foundations via Sequential Monte Carlo." In: *Syst. Biol.* 67.3 (2018), pp. 503–517.

References III

- A. Doucet, S. Godsill, and C. Andrieu. "On sequential Monte Carlo sampling methods for Bayesian filtering." In: *Stat. Comput.* 10.3 (2000), pp. 197–208.
- J. Duchi, E. Hazan, and Y. Singer. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization." In: *J. Mach. Learn. Res.* 12 (2011), pp. 2121–2159.
- R. Elliott. "Exact adaptive filters for Markov chains observed in Gaussian noise." In: *Automatica* 30.9 (1994), pp. 1399–1408.
- G. Evensen. "Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics." In: *J. Geophys. Res: Oceans* 99 (1994), pp. 10143–10162.
- A. Gelb. *Applied Optimal Estimation*. MIT Press, 1974.
- V. Gilja, C. Pandarinath, C. H. Blabe, P. Nuyujukian, J. D. Simeral, A. A. Sarma, B. L. Sorice, J. A. Perge, B. Jarosiewicz, L. R. Hochberg, K. V. Shenoy, and J. M. Henderson. "Clinical translation of a high-performance neural prosthesis." In: *Nat. Med.* 21.10 (2015), pp. 1142–1145.
- X. Glorot and Y. Bengio. "Understanding the difficulty of training deep feedforward neural networks." In: *Int. Conf. Artif. Intell. Stat.* Vol. 9. 2010, pp. 249–256.
- G. H. Golub. "Some Modified Matrix Eigenvalue Problems." In: *SIAM Rev. Soc. Ind. Appl. Math.* 15.2 (1973), pp. 318–334.
- M. Gönen and E. Alpaydın. "Multiple Kernel Learning Algorithms." In: *J. Mach. Learn. Res.* 12 (2011), pp. 2211–2268.
- N. J. Gordon, D. J. Salmond, and A. F. M. Smith. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation." In: *IEE Proc. F - Radar Signal Process.* 140.2 (1993), pp. 107–113.
- E. C. Hall. *Case History of the Apollo Guidance Computer*. MIT, 1966.

References IV

- J. M. Hammersley and K. W. Morton. “Poor Man’s Monte Carlo.” In: *J. Roy. Stat. Soc. Ser. B (Stat. Methodol.)* 16.1 (1954), pp. 23–38.
- D. G. Hoag. “Apollo Navigation, Guidance, and Control Systems: A Progress Report.” In: *National Space Meeting of the Institute of Navigation*. 1969.
- S. Hochreiter and J. Schmidhuber. “Long Short-Term Memory.” In: *Neural Comput.* 9.8 (1997), pp. 1735–1780.
- B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning System: Theory and Practice*. 5th. Springer, 2001.
- B. R. Hunt, E. J. Kostelich, and I. Szunyogh. “Efficient data assimilation for spatiotemporal chaos: A local ensemble transform Kalman filter.” In: *Physica D* 230.1 (2007), pp. 112–126.
- S. Ioffe and C. Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.” In: *Int. Conf. Mach. Learn.* Vol. 37. 2015, pp. 448–456.
- K. Ito. “Gaussian filter for nonlinear filtering problems.” In: *IEEE Conf. Decis. Control*. Vol. 2. 2000.
- K. Ito and K. Xiong. “Gaussian filters for nonlinear filtering problems.” In: *IEEE Trans. Autom. Control* (2000), pp. 910–927.
- B. Jarosiewicz, N. Y. Masse, D. Bacher, S. S. Cash, E. Eskandar, G. Friehs, J. P. Donoghue, and L. R. Hochberg. “Advantages of closed-loop calibration in intracortical brain-computer interfaces for people with tetraplegia..” In: *J. Neural Eng.* 10.4 (2013).

References V

- B. Jarosiewicz, A. A. Sarma, D. Bacher, N. Y. Masse, J. D. Simeral, B. Sorice, E. M. Oakley, C. Blabe, V. Pandarinath C.and Gilja, S. S. Cash, E. N. Eskandar, G. M. Friehs, J. M. Henderson, K. V. Shenoy, J. P. Donoghue, and L. R. Hochberg. "Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface." In: *Sci. Transl. Med.* 7.313 (2015), pp. 1–11.
- S. J. Julier and J. K. Uhlmann. "New extension of the Kalman filter to nonlinear systems." In: *Proc. SPIE* 3068 (1997), pp. 182–193.
- R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems." In: *J. Basic Eng* 82.1 (1960), pp. 35–45.
- R. E. Kalman and R. S. Bucy. "New Results in Linear Filtering and Prediction Theory." In: *J. Basic Eng.* 83.1 (1961), pp. 95–108.
- H. Kushner. "Approximations to optimal nonlinear filters." In: *IEEE Trans. Autom. Control* 12.5 (1967), pp. 546–556.
- R. van der Merwe. "Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models." PhD thesis. Oregon Health & Science University, 2004.
- P. del Moral. "Nonlinear Filtering Using Random Particles." In: *Theory Probab. Appl* 40.4 (1996), pp. 690–701.
- È. A. Nadaraya. "On a regression estimate." In: *Teor. Verojatnost. i Primenen.* 9 (1964), pp. 157–159.
- C. Olah. *Chris Olah's blog: Understanding LSTM Networks*. 2015. URL: `http://colah.github.io`.
- A. L. Orsborn, H. G. Moorman, S. A. Overduin, M. M. Shanechi, D. F. Dimitrov, and J. M. Carmena. "Closed-loop decoder adaptation shapes neural plasticity for skillful neuroprosthetic control." In: *Neuron* 82.6 (2014), pp. 1380–1393.
- E. Ott, B. R. Hunt, I. Szunyogh, A. V. Zimin, E. J. Kostelich, M. Corazza, E. Kalnay, D. J. Patil, and J. A. Yorke. "A local ensemble Kalman filter for atmospheric data assimilation." In: *Tellus A* 56.5 (2004), pp. 415–428.

References VI

- J. A. Perge, M. L. Homer, W. Q. Malik, S. Cash, E. Eskandar, G. Friehs, J. P. Donoghue, and L. R. Hochberg. “Intra-day signal instabilities affect decoding performance in an intracortical neural interface system.” In: *J. Neural Eng* 10.3 (2013).
- J. A. Perge, S. Zhang, W. Q. Malik, M. L. Homer, S. Cash, G. Friehs, E. N. Eskandar, J. P. Donoghue, and L. R. Hochberg. “Reliability of directional information in unsorted spikes and local field potentials recorded in human motor cortex.” In: *J. Neural Eng.* 11.4 (2014).
- V. Pham, T. Bluche, C. Kermorvant, and J. Louradour. “Dropout Improves Recurrent Neural Networks for Handwriting Recognition.” In: *Int. Conf. Front. Hand.* 2014, pp. 285–290.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.
- M. M. Shanechi, A. L. Orsborn, and J. M. Carmena. “Robust Brain-Machine Interface Design Using Optimal Feedback Control Modeling and Adaptive Point Process Filtering.” In: *PLoS Comput. Biol.* 12.4 (2016), pp. 1–29.
- M. M. Shanechi, A. Orsborn, H. Moorman, S. Gowda, and J. M. Carmena. “High-performance brain-machine interface enabled by an adaptive optimal feedback-controlled point process decoder.” In: *Conf. Proc. IEEE Eng. Med. Biol. Soc.* (2014), pp. 6493–6496.
- L. Shpigelman, H. Lalazar, and E. Vaadia. “Kernel-ARMA for Hand Tracking and Brain-Machine Interfacing During 3D Motor Control.” In: *Adv. Neural Inf. Process. Syst.* (2008), pp. 1489–1496.
- N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting.” In: *J. Mach. Learn. Res.* 15 (2014), pp. 1929–1958.

References VII

- S. Suner, M. R. Fellows, C. Vargas-Irwin, G. K. Nakata, and J. P. Donoghue. “Reliability of signals from a chronically implanted, silicon-based electrode array in non-human primate primary motor cortex.” In: *IEEE T Neur Sys Reh* 13.4 (2005), pp. 524–541.
- D. Sussillo, P. Nuyujukian, J. M. Fan, J. C. Kao, S. D. Stavisky, S. Ryu, and K. Shenoy. “A recurrent neural network for closed-loop intracortical brain–machine interface decoders.” In: *J. Neural Eng.* 9.2 (2012).
- D. Sussillo, S. D. Stavisky, J. C. Kao, S. I. Ryu, and K. V. Shenoy. “Making brain–machine interfaces robust to future neural variability.” In: *Nat. Commun.* 7 (2016).
- D. Thompson, L. R. Quitadamo, L. Mainardi, K. u. R. Laghari, S. Gao, P.-J. Kindermans, J. D. Simeral, R. Fazel-Rezai, M. Matteucci, T. H. Falk, L. Bianchi, and J. E. Chestek C. A.and Huggins. “Performance measurement for brain–computer or brain–machine interfaces: a tutorial.” In: *J. Neural Eng.* 11.3 (2014).
- A. W. van der Vaart. *Asymptotic statistics*. Vol. 3. Cambridge University Press, 1998.
- E. A. Wan and R. van der Merwe. “The unscented Kalman filter for nonlinear estimation.” In: *Adaptive Syst. for Signal Process., Commun., and Control Symp.* 2000, pp. 153–158.
- L. Wang, A. Bouchard-Côté, and A. Doucet. “Bayesian Phylogenetic Inference Using a Combinatorial Sequential Monte Carlo Method.” In: *J. Am. Stat. Assoc.* 110.512 (2015), pp. 1362–1374.
- G. S. Watson. “Smooth regression analysis.” In: *Sankhyā Ser. A* 26 (1964), pp. 359–372.
- W. Zaremba, I. Sutskever, and O. Vinyals. “Recurrent Neural Network Regularization.” In: *ArXiv e-prints* (2014).
- M. D. Zeiler. “ADADELTA: An Adaptive Learning Rate Method.” In: *ArXiv e-prints* (2012).