Active Learning

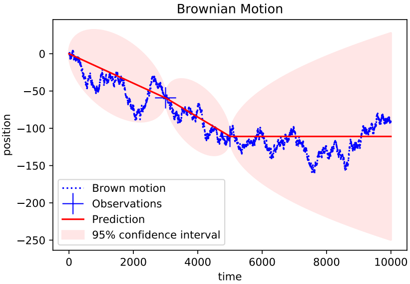
- You havee 'easy to get' unlabeled data.
- The evaluation of data is expensive.
- The task is to select the next data sample to evaluate.

scikit-activeml

- Gaussian Processes coupled with Bayesian optimization may be viewed as a special case for active learning.

Gaussian Processes

- An infinite (continuous) number of Gaussian variables
- to any value x a new variable $N(\mu = f(x), \Sigma_{x|rest})$
- we have only a finite number of observations which means a finite number of variables
    - we can marginalize unobserved variables out (the integral is 1, we multiply by 1, we just remove),
- we can predict at any x, continuously.



Gaussian Processes

C. E. Rasmussen & C. K. I. Williams, Gaussian Processes for Machine Learning, the MIT Press, 2006

Definition (Gaussian Process)

A Gaussian process is a set of random variables where any finite subset follows multivariate Gaussian distribution.

We define the mean $m(x)$ and the symmetric positive semidefinite covariance function $k(x, x^{\prime})$:

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})] \\ k(\mathbf{x}, \mathbf{x}^{\prime}) &= \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}^{\prime}) - m(\mathbf{x}^{\prime}))] \end{aligned}$$

a Gaussian process is

$$f(\mathbf{x}) = \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}^{\prime})).$$

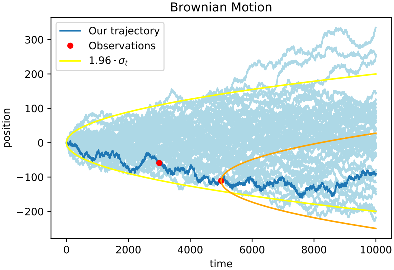
We assume $m(\mathbf{x}) = \mathbf{0}$ it simplifies the formulas.

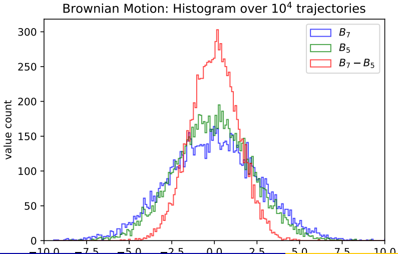
Brownian Motion (Wiener Process)

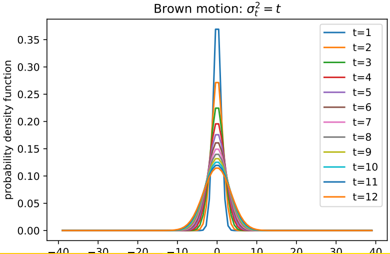
https://www.coursera.org/lecture/stochasticprocesses/week-4-6-two-definitions-of-a-brownian-motion-THRqL

Definition (Brownian motion 1)

- $B_0 = 0$ for sure
- stationary and independent increments
- $B_s - B_t \sim N(0, s - t)$







Definition (Brownian motion 1)

- $B_0 = 0$ almost surely
- B_t stationary and independent increments
- $B_s - B_t \sim N(0, s-t)$

Definition (Brownian motion 2)

Gaussian process with

- $m = 0$ and
- $k(x, x') = min(x, x')$.

Positive semidefinite:

- $min(t, s) = \int_0^\infty f_t(x) f_s(x) dx$
- $f_t(x) f_s(x) = 1$ iff $x \in [0, t] \& x \in [0, s]$

Lemma (2⇒1)

- $K(0,0) = min(0,0) = 0$
- *The process has variance 0 at* $t = 0$ *and* $m(0) = 0$.
- *covariance is linear in both arguments,* $s \geq t$

$$\begin{aligned} cov(B_s - B_t, B_s - B_t) &= cov(B_s, B_s) - cov(B_t, B_s) - cov(B_s, B_t) + cov(B_t, \\ &= s - 2t + t = s - t \end{aligned}$$

- *increments,* $s \geq t \geq b \geq a$ *# independence skipped, from Gaussian vectors*

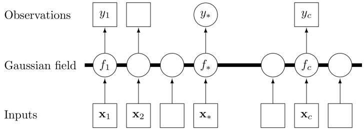
$$\begin{aligned} cov(B_b - B_a, B_s - B_t) &= cov(B_b, B_s) - cov(B_a, B_s) - cov(B_b, B_t) + cov(B_a, B_t) \\ &= b - a - b + a = 0 \end{aligned}$$

Normal Distribution

Definition (Brownian motion 2)

Gaussian process with

- $m = 0$ and
- $k(x, x') = min(x, x')$.



- The covariance on $\mathbf{y}$ is defined by the covariance on the inputs $\mathbf{x}$.
- the covariance defines also the distribution on functions f:

$$\mathbf{f}_* \sim N(\mathbf{0}, K(X_*, X_*)).$$

- Without noise, we observe $\mathbf{y}$ and we want to predict $\mathbf{f}_*$:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X, X) & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right)$$

X=[3,7]
y^T=[0.5, 1.11]
K(xs,X)=[min(xs,a) for a in X]

$$K(X, X) = \begin{bmatrix} min(3, 3) & min(3, 7) \\ min(3, 7) & min(7, 7) \end{bmatrix} = \begin{bmatrix} 3 & 3 \\ 3 & 7 \end{bmatrix}$$

Prediction

- noisy-free observations $y = f(\mathbf{x})$

$$cov(y_p, y_q) = k(\mathbf{x}_p, \mathbf{x}_q)$$

- noisy observations $y = f(\mathbf{x}) + \epsilon$, $\epsilon \sim N(0, \sigma_n^2)$

$$\begin{aligned} cov(y_p, y_q) &= k(\mathbf{x}_p, \mathbf{x}_q) + \sigma_n^2 \delta_{pq} \\ cov(\mathbf{y}) &= K(X, X) + \sigma_n^2 I \end{aligned}$$

- We observe $\mathbf{y}$ and we want to predict $\mathbf{f}_*$:

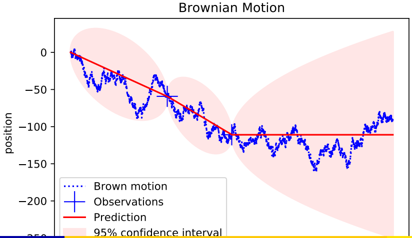
$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right)$$

- Predictive distribution

$$\begin{aligned} \mathbf{f}_* | X, \mathbf{y}, X_* &\sim N(\bar{\mathbf{f}}_*, cov(\bar{\mathbf{f}}_*)) \\ \bar{\mathbf{f}}_* &\triangleq \mathbb{E}[\mathbf{f}_* | X, \mathbf{y}, X_*] = K(X_*, X)[K(X,X) + \sigma_n^2 I]^{-1} \mathbf{y} \\ cov(\mathbf{f}_*) &= K(X_*, X_*) - K(X_*, X)[K(X_*, X) + \sigma_n^2 I]^{-1} K(X, X_*) \end{aligned}$$

$$\mathbf{f}_*|X,\mathbf{y},X_* \sim N(\bar{\mathbf{f}}_*, cov(\bar{\mathbf{f}}_*))$$
$$\bar{\mathbf{f}}_* \triangleq \mathbb{E}[\mathbf{f}_*|X,\mathbf{y},X_*] = K(X_*,X)[K(X,X)+\sigma_n^2 I]^{-1}\mathbf{y}$$
$$cov(\mathbf{f}_*) = K(X_*,X_*) - K(X_*,X)[K(X_*,X)+\sigma_n^2 I]^{-1}K(X,X_*)$$

$$\bar{\mathbf{f}}_* \triangleq \mathbb{E}\left[\mathbf{f}_*|\begin{bmatrix}3\\7\end{bmatrix},\begin{bmatrix}0.5\\11\end{bmatrix},[4]\right] = [3,4]\begin{bmatrix}3+\sigma_n^2 & 3\\ 3 & 7+\sigma_n^2\end{bmatrix}^{-1}\begin{bmatrix}0.5\\11\end{bmatrix}$$
$$cov(\mathbf{f}_*) = min(4,4) - [3,4]\begin{bmatrix}3+\sigma_n^2 & 3\\ 3 & 7+\sigma_n^2\end{bmatrix}^{-1}\begin{bmatrix}3\\4\end{bmatrix}$$

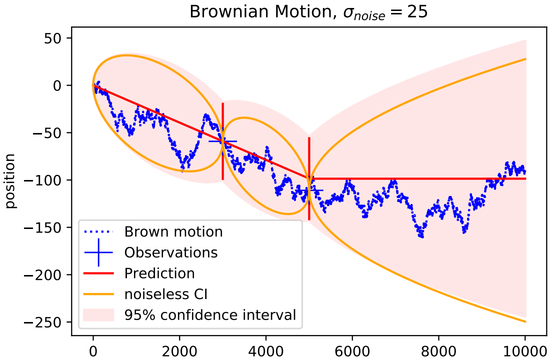


Predictive distribution

Prediction

- is a linear function of observations **y**
  - for $\alpha \Leftarrow (K + \sigma_n^2 I)^{-1}\mathbf{y}$
  - we predict $\overline{f}(\mathbf{x}_*) \Leftarrow \sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x}_*)$

- The red vertical bars show the variance due to the observation noise.



Definition (First Set of Kernel Functions)

- **Radial Basis Function (RBF)** covariance function with the **length scale** parameter ℓ is defined

$$cov(f(\mathbf{x}_p), f(\mathbf{x}_q)) = RBF(\mathbf{x}_p, \mathbf{x}_q) = \exp\left(-\frac{1}{2}\frac{|\mathbf{x}_p - \mathbf{x}_q|^2}{\ell^2}\right).$$

- **Constant** covariance function with the **constant** parameter is defined

$$cov(f(\mathbf{x}_p), f(\mathbf{x}_q)) = Constant(\mathbf{x}_p, \mathbf{x}_q) = constant.$$

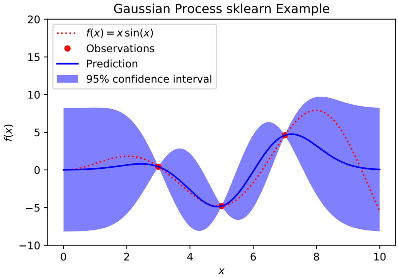
- **Squared exponential (SE)** covariance function with hyperparameters ℓ^2 lenghtscale and σ_f^2 signal variance

$$\begin{aligned} k(\mathbf{x}_p, \mathbf{x}_q) &= \sigma_f^2 \exp\left(-\frac{1}{2}\frac{|\mathbf{x}_p - \mathbf{x}_q|^2}{\ell^2}\right) \\ &= Constant(\mathbf{x}_p, \mathbf{x}_q) * RBF(\mathbf{x}_p, \mathbf{x}_q) \end{aligned}$$

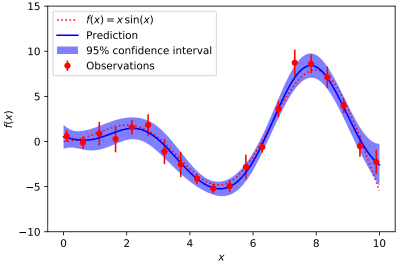
- can be defined as a **product kernel** of the Constant and RBF kernels.
- There is also a **sum kernel** kernel function $+$.

Scikitlearn Examples

- Noiseless observations.



- The red vertical bars show the variance due to the observation noise.



- The parameters may be fitted by the gradient update.
- The observation noise *alpha* may be specific for each observation (right), identically 0 (left) or constant.

```
kernel = C(1.0, (1e-3, 1e3)) * RBF(10, (100e-2, 100e2))
gp = GaussianProcessRegressor(kernel=kernel, alpha=dy ** 2)
gp.fit(X, y)
```

Marginal likelihood

- The parameters may be automatically tuned by gradiently maximize the marginal likelihood.
- 'In sample' prediction $\mathbf{f}$ follows: $\mathbf{f} \sim N(\mathbf{0}, K(X,X))$.

Lemma

The marginal log likelihood is

- *for noisy-free observations* $\mathbf{y} = \mathbf{f}$*:*

$$\log p(\mathbf{y}|X) = \log p(\mathbf{f}|X) = -\frac{1}{2}\mathbf{f}^T K^{-1}\mathbf{f} - \frac{1}{2}\log|K| - \frac{N}{2}\log 2\pi$$

- *For noisy observations* $\mathbf{y}|\mathbf{f} \sim N(\mathbf{f}, \sigma_n^2 I)$*,* $\mathbf{y} \sim N(0, K + \sigma_n^2 I)$

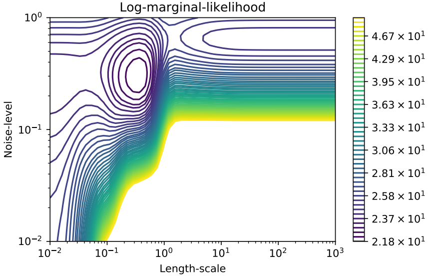
$$\log p(\mathbf{y}|X) = -\frac{1}{2}\mathbf{y}^T (K + \sigma_n^2 I)^{-1}\mathbf{y} - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{N}{2}\log 2\pi.$$

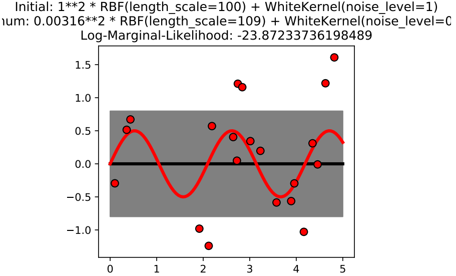
The noise level may be tuned as well by (sum)adding the **WhiteKernel**. **WhiteKernel**= *noise_level* iff we address the same variable $(\mathbf{x}_p, \mathbf{x}_p)$, otherwise *WhiteKernel* = 0

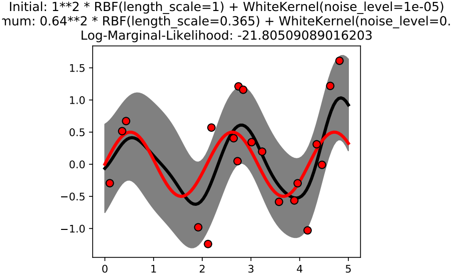
Hyperparameter Fit

Scikitlearn example:

- The log-marginal function has two local maxima.
- The log-marginal maxima corresponds to the two models.







Definition (Further Kernel Functions)

- **ExpSineSquared** kernel function with the parameters **length scale** ℓ and the **periodicity** $p > 0$ (d is the distance) is defined

$$cov(f(\mathbf{x}_q), f(\mathbf{x}_r)) = \exp\left(-\frac{2\sin^2(\pi d(\mathbf{x}_q, \mathbf{x}_r)/p)}{\ell^2}\right).$$

  Usefull for periodic functions.

- **Dot product** kernel function with the **inhomogenicity** parameter σ_0 is defined

$$cov(f(\mathbf{x}_p), f(\mathbf{x}_q)) = \sigma_0 + \mathbf{x}_p \cdot \mathbf{x}_q.$$

  Useful to capture the trend, often combined with exponential kernel.

- **Rational Quadratic** kernel function with hyperparameters ℓ^2 lenghtscale and mixture α

$$k(\mathbf{x}_p, \mathbf{x}_q) = \left(1 + \frac{d(\mathbf{x}_p, \mathbf{x}_q)^2}{2\alpha\ell^2}\right)^{-\alpha}$$

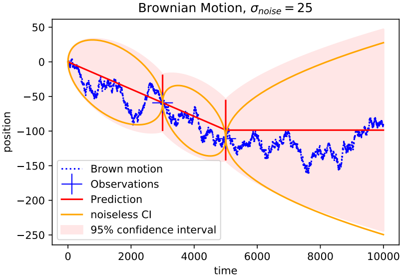
  The mixure of many RBF kernel lengthscales.

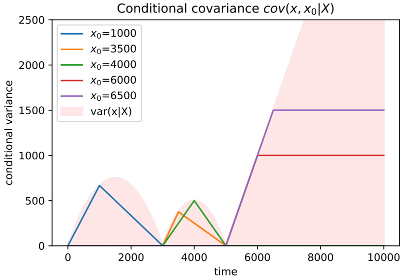
Conditional Covariance

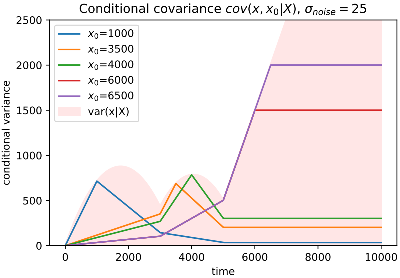
- Consider the conditional covariance, the relation of two unobserved points x and x_0.

left Noiseless Brown motion example. The covariance is zero outside the x_0 closest observations interval.

right Brown motion with a high noise level.

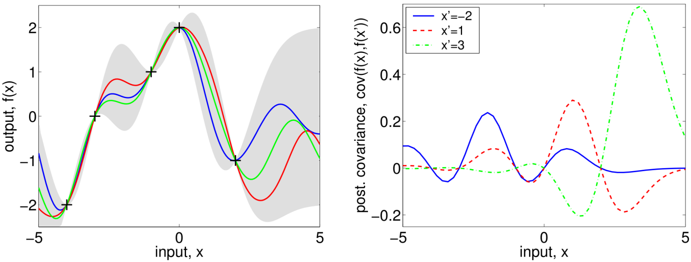






Conditional Covariance Rassmussen Example

- The conditional covariance may be also negative.
- Most kernels have a continuous first derivative. This makes the conditional covariance negative with points on the other side of the closest observation.



(a), posterior (b), posterior covariance

Matérn

- Most kernel function have many derivatives.
- The Matérn kernel $\nu = 1.5$ ('nu') has only the first derivative. It is able to model less smooth functions.
- As $\nu \to \infty$, it becomes a RBF kernel.

Definition (Matérn kernel)

The **Matérn** kernel with parameters $\nu = k + \frac{1}{2}$ and ℓ is defined

$$k(\mathbf{x}_p, \mathbf{x}_q) = \frac{1}{2^{\nu-1}\Gamma(\nu)} \left(\frac{\sqrt{2\nu}}{\ell} d(\mathbf{x}_p, \mathbf{x}_q) \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}}{\ell} d(\mathbf{x}_p, \mathbf{x}_q) \right)$$

The **Modified Bessel functions** (for α not integer, the limit otherwise) are defined

- $I_\alpha(x) = \sum_{m=0}^{\infty} \frac{1}{m!\Gamma(m+\alpha+1)} \left(\frac{x}{2}\right)^{2m+\alpha}$
- $K_\alpha(x) = \frac{\pi}{2} \frac{I_{-\alpha}(x) - I_\alpha(x)}{sin \alpha \pi}$.

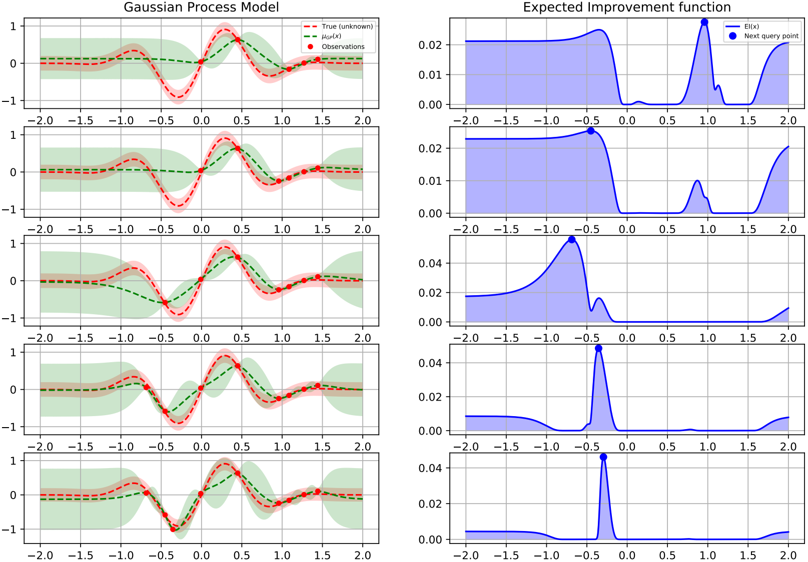
Bayesian Optimization

- Bayesian Optimization is used when
  - We are solving: $x^* = \arg\min_x f(x)$
  - $f(x)$ is a black box function
  - f is expensive to evaluate
  - the evaluations may be noisy.
- If any condition is not true, a better algorithm exists.
- We search the point $\mathbf{x}$ to observe.
- scikit-optimize = skopt Python package
- we minimize $\mathbf{y}$ and search the maximal probability of improvement
- 'the chance to improve' is expressed by the **Expected improvement** (EI)

Bayesian Optimization Algorithm

- Evaluate $\mathbf{y}$ on X, let $\mathbf{y} = y(X)$ and calculate conditional means and covariances
- repeat forever
  - $x^{new} = argmax_x EI(x)$ add x into X
  - Evaluate $\mathbf{y} = y(x)$ and add y to $\mathbf{y}$.
  - re–estimate the Gaussian process (the parameters of the covariance).

Bayesian Optimization Example [Skopt]



Expected Improvement Aquisition Function

- we search the point $\mathbf{x}$ to observe
- we minimize y, we already have the training data X,$\mathbf{y}$
- the search the maximal probability of improvement is expressed by the **Expected improvement** (EI)

$$\begin{aligned} EI(x) &= \mathbb{E}[(min(Y(X)) - Y(x))^{+} | Y(X) = \mathbf{y}] \\ &= \mathbb{E}[(min(\mathbf{y}) - Y(x))^{+} | Y(X) = \mathbf{y}] \end{aligned}$$

this can be solved analytically (Φ cummulative df, ϕ pdf Gaussian distribution):

$$EI(x) = (min(\mathbf{y}) - \mu(x))\Phi\left(\frac{min(\mathbf{y}) - \mu(x)}{\sigma(x)}\right) + \sigma(x)\phi\left(\frac{min(\mathbf{y}) - \mu(x)}{\sigma(x)}\right)$$

to maximize $\mathbf{y}$:

$$EI(x) = (\mu(x) - max(\mathbf{y}))\Phi\left(\frac{\mu(x) - max(\mathbf{y}) - \xi}{\sigma(x)}\right) + \sigma(x)\phi\left(\frac{\mu(x) - max(\mathbf{y}) - \xi}{\sigma(x)}\right).$$

- if 'xi' $\xi > 0$ we ignore small improvements.

Paralelization: The Constant Liar

$$
\begin{aligned}
EI(x) &= \mathbb{E}[(min(Y(X)) - min(Y(x^{(n+1)}), Y(x^{(n+2)}), \ldots, Y(x^{(n+k)})))^{+} | Y(X) = \mathbf{y} \\
&= \mathbb{E}[(min(\mathbf{y}) - Y(x))^{+} | Y(X) = \mathbf{y}]
\end{aligned}
$$

it does not have direct formula. It is solved by Markov Chain simulation.

- We estimate the observations y by an estimate (min, max, mean)
- and run the evaluation in parallel.

That means the covariance is correctly estimated, the mean must be corrected later.

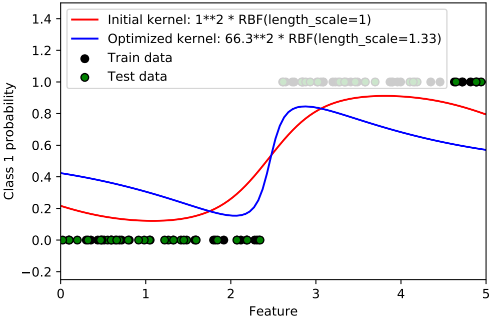
ParBayesianOptimization R package

Definition (Other Aquisition Functions)

- Probability of Improvement: $PI(f(x_*) < min(\mathbf{y})) = \Phi\left(\frac{min(\mathbf{y}) - \mu(x_*)}{\sigma(x_*)}\right)$
- Lower Confidence Bound: $LCB(x) = \mu(x) - \kappa \cdot \sigma(x)$.

GP for Classification

- GP for classification are more complex and 'only an approximation'
- still, it is worth to try the `sklearn.gaussian_process.GaussianProcessClassifier` .
- We estimate a latent function f as before
- we link it to $\langle 0, 1 \rangle$ interval by the sigmoid function (or Φ).
- The log-marginal-likelihood does not have a closed analytical form anymore.
- can be approximated by Hessian matrix, the algorithm works in $O(N^3)$, not too bad.

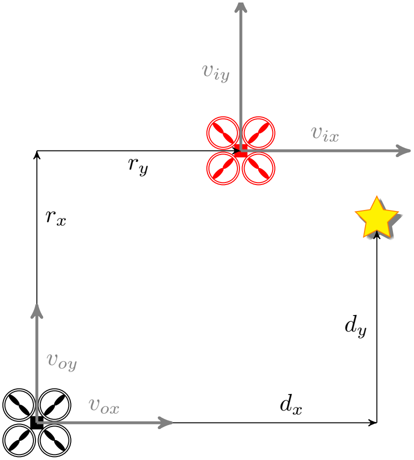


POMDP Applications

- Karkus, Hsu, Lee: **QMDP-Net: Deep Learning for Planning under Partial Observability** https://proceedings.neurips.cc/paper/2017/file/e9412ee564384b987d086df32d4 Paper.pdf
- Eric Mueller and Mykel J. Kochenderfer :**Multi-Rotor Aircraft Collision Avoidance using Partially Observable Markov Decision Processes**, American Institute of Aeronautics and Astronautics https://aviationsystemsdivision.arc.nasa.gov/publications/2016/AIAA-2016-3673.pdf

POMDP Aircraft Collision Avoidance

- the algorithms designed for fixed-wing aircraft analyze
  - turns
  - vertical meneuvers
- multirotor aircraft (drones) and helicopters can also
  - horizontal plane accelerations
- state 2D, (3D)
  - relative range states r_x, r_y, (r_z)
  - velocities for the ownship v_{ox}, v_{oy}, (v_{oz})
  - velocities for the intruder v_{ix}, v_{iy}, (v_{iz})
  - absolute displacement from the desired trajectory d_x, d_y, (d_z)
  - the desired trajectory is normalized to unit velocity in the $x$axis and zero velocity in the y axis.



MDP Transitions

- the prediction horizon is very short
- updates done every 0.1 to 1 seconds
- simple update equation are sufficient
- not a benefit to using more complex dynamic equations.
- a_x, a_y acceleration by the ownship
- N_* noise to the ownship, intruder, x and y axis
    - $N_o(\mu = 0, 0.30s^{-2})$, $N_i(\mu = 0, 0.45s^{-2})$,
- Bellman update

  transition from s with acceleration a to $s^|$

$$Q[s, a] \leftarrow R(s, a) + \gamma \sum_{s^|} T(s^| | s, a) max_{a^|} Q[s^|, a^|].$$

$$\begin{aligned}
\dot{r}_x &= v_{ix} - v_{ox} \\
\dot{r}_y &= v_{iy} - v_{oy} \\
\dot{v}_{ox} &= a_x + N_{ox} \\
\dot{v}_{oy} &= a_y + N_{oy} \\
\dot{v}_{ix} &= N_{ix} \\
\dot{v}_{iy} &= N_{iy} \\
\dot{d}_x &= v_{tx} - v_{ox} \\
\dot{d}_y &= v_{ty} - v_{oy}
\end{aligned}$$

Reward

- Minimum reward R_{min}
  - collision
  - physically impossible states
  - keeps the sum finite
- we prefer no acceleration
- we prefer long distance to the intruder
- we prefer short distance to the desired trajectory
- K_s, K_T, R_{min} weights was learned, k weights was $= 1$.

$$R(s,a) = max\left[R_{min}, -(k_{ax}|a_x| + k_{ay}|a_y|) - K_s\frac{1}{k_{rx}r_x^2 + k_{ry}r_y^2} - K_T(k_{dx}d_x^2 + k_{dy}d_y^2)\right]$$

QMDP Approximation

- offline optimization
  - a few hours for coarse discretization, 1 PC
  - initially stationary intruders
  - intruders moving at uniform velocity with a variety of relative headings angles
  - intruders state and dynamic uncertainty were added to the encounters.
- all values normalized
  - the coarse set contained a total of 765,625 discrete states
  - the finely discretized version contained 9,529,569 states.

State variable	State Description	Discretization
r_x, r_y	Intruder range components	$-15, [-7, -3], -1, 0, 1, [3, 7], 15$
v_{ox}, v_{oy}	ownship velocity components	$-5, -3, -1, 0, 1, 3, 5\ s^{-1}$
v_{ix}, v_{iy}	intruder velocity components	$-5, [-3], -1, 0, 1, [3], 5\ s^{-1}$
d_x, d_y	desired trajectory distance	$-10, [-3], -1, 0, 1, [3], 10$

Evaluation Function

- The primary goal is to remain safely separated from the intruder aircraft.
  - $r_{5\%CPA}$ 'the closest point of approach', we allow 5% trajectories a little bit closer.
- Figure: required 1.5 units, never closer than 1.1 units.
- Mean deviation distance from the desired trajectory μ_{dev}.

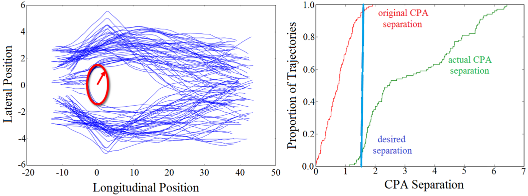
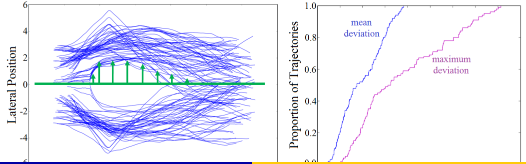


Figure 3: Separation metric used to evaluate the collision avoidance algorithm



Reward Tuning – Bayesian Optimization

- We tune $R_P = (K_T, K_s, R_{min})$
- β weights the two objective functions

$$F(R_P) = (\beta \times (r_{5\%CPA})^{-1} + (1-\beta) \times \mu_{dev}).$$

- Gaussian process models $F(R_P)$.
- We determine the point at which the objective function is expected to have the largest improvement, $E[I(F(R_P))]$ over that of the current minimum.
- This set of R_P is passed to QMDP to evaluate.
- until convergence.

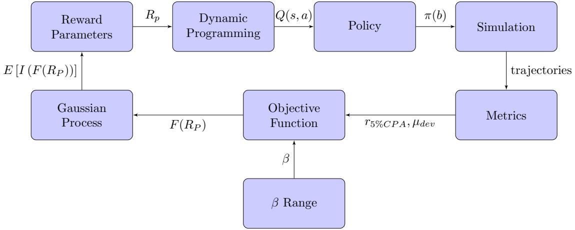
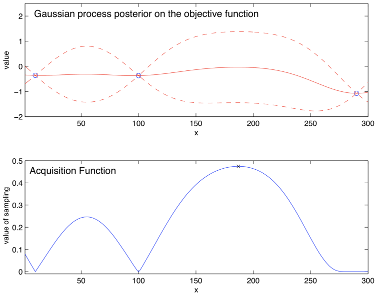


Figure 5: Process for tuning POMDP reward parameters

Bayesian Optimization

- we know QMDP and F values for one or more $\mathbf{x} = R_P$ points
- we search the point $x = R_P^*$ to observe
- we minimize $y = F(R_P^*)$ and search the maximal probability of improvement
- 'the chance to improve' is expressed by the **Expected improvement** (EI)

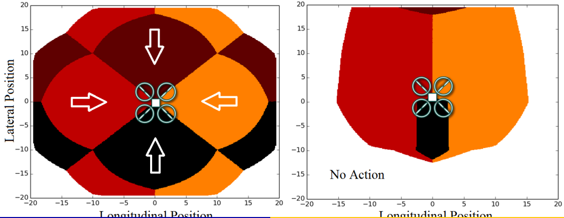


Peter I. Frazier: A Tutorial on Bayesian Optimization, rXiv:1807.02811v1 [stat.ML] 8 Jul 2018

Value Iteration, QMDP Policies

- considerable improvement in convergence speed by initializing by the value of previously evaluated policy
- the value of maximum negative reward influenced the convergence speed
- $\gamma \leftarrow 0.99$ taking hundred iterations to converge.
- smaller γ did not ensure the return to the desired path.

- Figures: Owhship at the origin
- different intruder positions
- policies indicated by color: black=up, red=right
- left: both own and intruder velocities are zero, $d = 0$
- right: owhship is moving in the positive y–axis direction at 1 s^{-1} with zero trajectory error and nominal trajectory matches the velocity.
- The intruder is stationary.



Beliefs

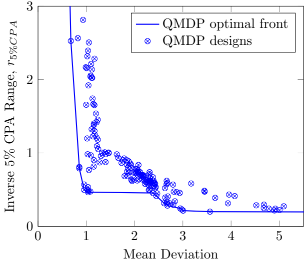
- Uncertainty does not increase with time
- State uncertainty is incorporated only when actions are selected
- a set of potential states is calculated from the observations received at each step.
- the potential states become the beliefs used to select an action.

$$\pi(b) = max_a \left[\sum_k Q(s^{(k)}, a) b^{(k)} \right]$$

- The value $Q(s^{(k)}, a)b^{(k)}$ approximated from QMDP solutions
  - rectangular interpolation between 2^n nearest neighbor
  - simplex interpolation between $n + 1$ nearest neighbor
  - prior work has found little benefit to using more sophisticated approaches.

Pareto Optimal frontier

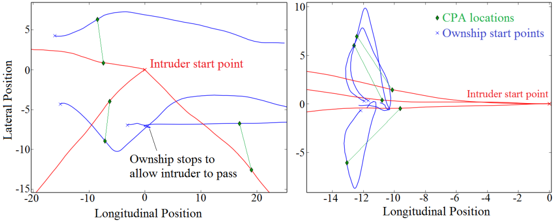
- 194 parameter sets evaluated
- β between 0.01 and 0.99 .
- resulting in nine non-dominated, Pareto–optimal designs.



Human Expert Check

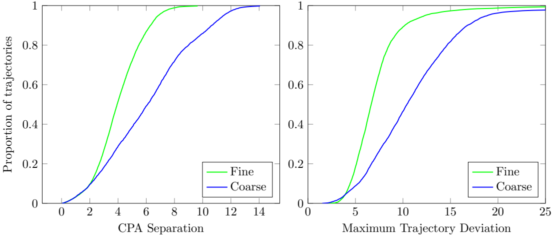
- Left: intruder starts at $(0, 0)$,
- random heading, fixed velocity of the intruder
- the ownship starts at the blue cross

- Right: The goal is hovering
- the intruder comes from the right with the unknown behaviour.



State Discretization

The fine discretization improves the results.



(a) CPA separations

(b) Maximum trajectory deviations

Figure 10: Cumulative distributions of encounter model metrics as a function of state discretization

Table of Contens