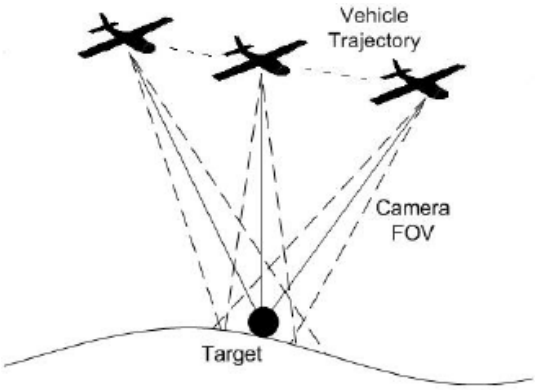
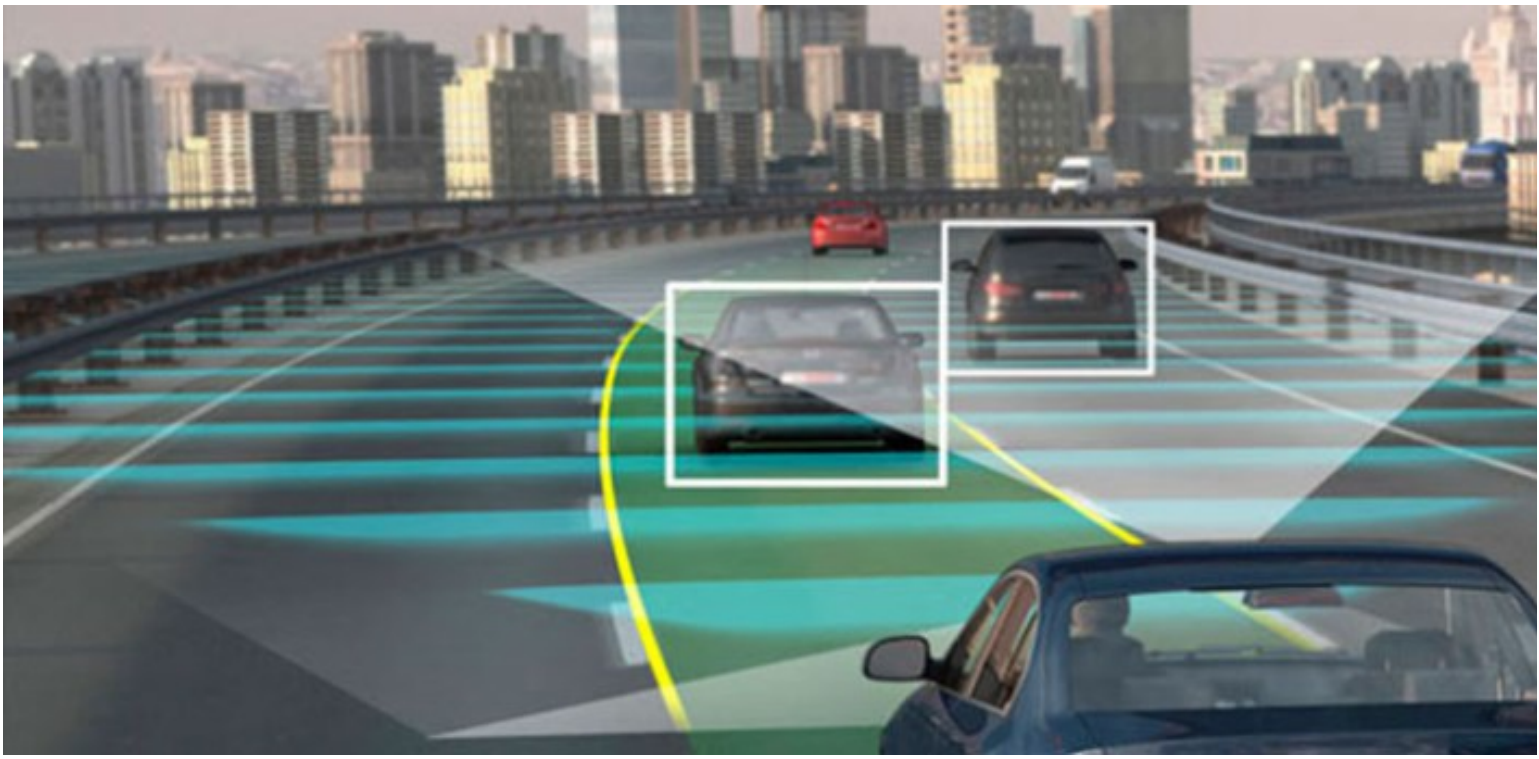
Part III. Bayesian Estimation

Localization: Given noisy measurements of a random variable x, how to estimate the location of x?



Tracking. Given noisy measurements of a random process x_k, how to estimate x_k? Self driving cars [Optimal Filtering]



Classical Bayesian Estimation

Model: obs y is a probabilistic function of random state x.

1. state $x \sim \pi_0$ where π_0 is a prior pdf/pmf
2. observation $y \sim p(y|x)$ (observation/sensor likelihood).

Aim: Given model π_0 and $p(y|x)$, estimate x given obs y.

4 Main Results:

1. Optimal estimate of state x given obs y is conditional mean:

$$\hat{x} \stackrel{\text{defn}}{=} \mathbb{E}\{x|y\}$$

CM is optimal in the sense of minimizing mean square error:

$$\mathbb{E}\{(\hat{x} - x)^2\} \leq \mathbb{E}\{(g(y) - x)^2\}$$

for any other estimator $g(y)$ of the state x.

2. Bayes rule:

$$\boxed{p(x|y) = \frac{\pi_0(x)p(y|x)}{\int_z \pi_0(z)p(y|z)dz}}$$

$$\text{posterior} \ \propto \ \text{prior} \ \times \ \text{likelihood}$$

3. The conditional mean estimate of x given observation y is

$$\boxed{\hat{x} = \mathbb{E}\{x|y\} = \int x\, p(x|y)dx}$$

4. The maximum aposteriori (MAP) estimate of x given y is:

$$x^{\text{MAP}} = \underset{x}{\text{argmax}}\, p(x|y) \qquad \text{(Bayesian classifier)}$$

x^{MAP} minimizes mis-classification prob $P(g(y) \neq x|y)$.

Example 1: Discrete rv case. Suppose $x \in \{1,2\}$, $y \in \{1,2\}$;

$\pi_0 = [0.2, 0.8]'$,

$P(y\|x)$	$y=1$	$y=2$
$x=1$	0.6	0.4
$x=2$	0.3	0.7

Suppose we observe $y = 1$. Compute Bayesian estimate of x.

$$P(x=1|y=1) = \frac{\pi_0(1)P(y=1|x=1)}{\pi_0(1)P(y=1|x=1)+\pi_0(2)P(y=1|x=2)} = \frac{1}{3}$$

$$P(x=2|y=1) = \frac{\pi_0(2)P(y=1|x=2)}{\pi_0(1)P(y=1|x=1)+\pi_0(2)P(y=1|x=2)} = \frac{2}{3}$$

MAP estimate is $x^{\text{MAP}} = \text{argmax}_x\, P(x|y=1) = 2$.
CM estimate is $\hat{x} = 1 \times 1/3 + 2 \times 2/3 = 1.667$.

Remarks: CM = MMSE = min var estimator

1. Conditional mean: soft estimate. MAP: hard estimate. MAP estimator = Bayes Classifier in machine learning.
2. $p(y|x)$ is "likelihood" - plausibility y was generated from x. Maximum likelihood estimate (MLE) is

$$\boxed{x^{\text{MLE}} = \operatorname*{argmax}_x p(y|x)}$$

   MLE does not use prior info. For above example $x^{\text{MLE}} = 1$.
3. If prior is non-informative i.e., $\pi_0(x)$ is constant wrt x, then $p(x|y) \propto p(y|x)$ and so MAP = MLE.
4. If likelihood is non-informative, i.e., $p(y|x)$ is indpt of x, then $p(x|y) = \pi_0(x)$. MLE is useless; $x^{\text{MAP}} = \text{argmax}_x\, \pi_0(x)$.

Proof of Optimality of Conditional Mean Estimator

Theorem: Suppose $\hat{x} \stackrel{\text{defn}}{=} \mathbb{E}\{x|Y\}$. Then for any estimator $g(Y)$,

$$\mathbb{E}\{(\hat{x}-x)^2\} \leq \mathbb{E}\{(g(Y)-x)^2\} \iff \operatorname*{argmin}_g \mathbb{E}\{(g(Y)-x)^2\} = \hat{x}.$$

Proof:

$\operatorname{argmin}_g \mathbb{E}\{(g(Y)-x)^2\} = \operatorname{argmin}_g \mathbb{E}\{(g(Y)-\hat{x}+\hat{x}-x)^2\} =$
$\operatorname{argmin}_g \mathbb{E}\{(g(Y)-\hat{x})^2\} + \underbrace{\mathbb{E}\{(\hat{x}-x)^2\}}_{\text{irrelevant}} + 2\underbrace{\mathbb{E}\{(g(Y)-\hat{x})'(\hat{x}-x)\}}_{0}.$

Last term is zero follows from law of iterated expectations.

$$\begin{aligned}\mathbb{E}\{(g(Y)-\hat{x})'(\hat{x}-x)\} &= \mathbb{E}\{\mathbb{E}\{(g(Y)-\hat{x})'(\hat{x}-x)|Y\}\} \\ &= \mathbb{E}\big\{(g(Y)-\hat{x})'\mathbb{E}\{(\hat{x}-x)|Y\}\big\} = 0\end{aligned}$$

Remark. Law of iterated expectation (LIE). $\mathbb{E}\{\mathbb{E}\{X|Y\}\} = \mathbb{E}\{X\}$. LIE is special case of *smoothing property of conditional expectations*: If $\mathcal{F} \subset \mathcal{G}$, then

$$\mathbb{E}\{\mathbb{E}\{X|\mathcal{G}\}|\mathcal{F}\} = \mathbb{E}\{X|\mathcal{F}\}$$

Special case: Choose $\mathcal{F} = \{\Omega, \emptyset\}$, then yields LIE

Example. Linear Discriminant Analysis

LDA = MAP estimator = $\operatorname{argmax}_x p(x|y)$

Has applications in

1. Face recognition
2. Pattern recognition

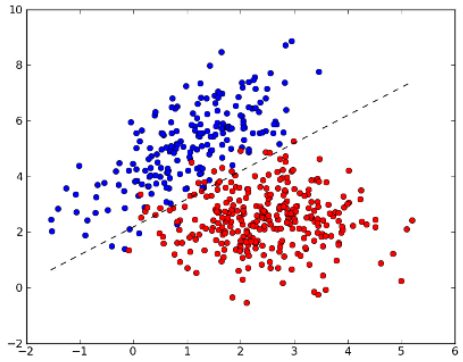
Example 1. Bayesian Classifier and Linear Discriminant Analysis (LDA)

Given data $y_1, \ldots, y_n$, classify into $x_1, \ldots, x_n$ where $x_i \in \{\text{spam} = 1, \text{ not spam} = 2\}$. So y is noisy observation of x.

Aim: Using training data construct optimal classifier $g(y)$ to minimize

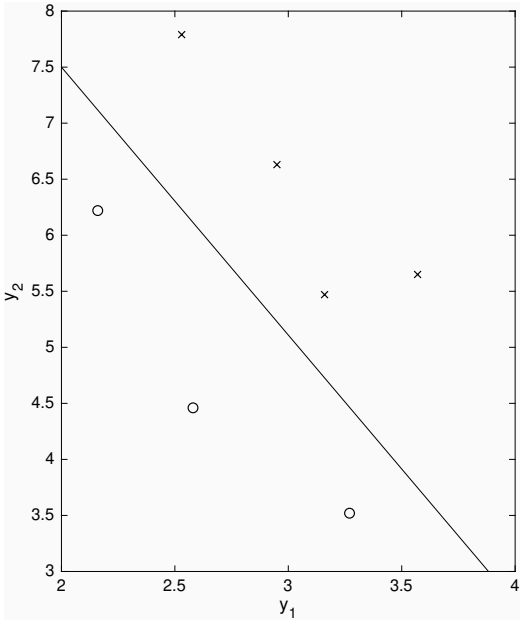
$$P(x \neq g(y)|y) = \text{prob of miss-classification}$$

Fig shows $y_i \in \mathbb{R}^2$ and linear classifier (in y)



Example: A factory produces optical lenses. Quality measured in terms of curvature and diameter. From quality control data:

Curvature $y(1)$	Diameter $y(2)$	Result	x
2.95	6.63	Passed	1
2.53	7.79	Passed	1
3.57	5.65	Passed	1
3.16	5.47	Passed	1
2.58	4.46	Not Passed	2
2.16	6.22	Not Passed	2
3.27	3.52	Not Passed	2



Aim: Use this training data to construct a Bayesian classifier.

Result: Bayes classifier that minimizes $P(x \neq g(y)|y)$ is MAP estimate

$$\boxed{g(y) = \arg\max_x P(x|y) = \arg\max_x \frac{P(y|x)P(x)}{\cancel{P(y)}}}$$

Proof: Let $u = g(y) \in \{1, 2\}$ = classification decision.

$$\text{Compute } u^* = \arg\min_u \mathbb{E}\{L(x,u)|y\} \text{ where } L(x,u) = \begin{cases} 1 & x \neq u \\ 0 & \text{otherwise} \end{cases}$$

$$\arg\min_u \mathbb{E}\{L(x,u)|y\} = \arg\min_u \sum_x L(x,u)P(x|y)$$

$$= \arg\min_u \sum_{x \neq u} P(x|y) = \arg\min_u (1 - P(u|y)) = \arg\max_u P(u|y)$$

Remark. Note MAP estimate g^* minimizes $P(x \neq g(y))$ since

$$\mathbb{E}\{I(x \neq g^*(y)|y\} \leq \mathbb{E}\{I(x \neq g(y)|y\} \forall y$$

$$\implies \mathbb{E}\{I(x \neq g^*(y)\} \leq \mathbb{E}\{I(x \neq g(y)\}$$

Since $X \in \{1, 2\}$, Bayes classifier is

$$g(y) = \begin{cases} 1 & P(x=1|y) > P(x=2|y) \\ 2 & \text{otherwise} \end{cases} = \begin{cases} 1 & \log \frac{P(y|1)}{P(y|2)} + \log \frac{P(1)}{P(2)} > 0 \\ 2 & \text{otherwise} \end{cases}$$

In machine learning, likelihood $P(y|x)$, prior $P(x)$ not known. Given training data $(x_1, \ldots, x_n, y_1, \ldots, y_n)$ how to build classifier?

1. Parameteric Classifier: assume Gaussian $P(y|x)$ and estimate parameters from data – linear discriminant anaylsis
2. Semi-parametric Classifier: assume a logistic model for posterior and estimate parameters from data

Method 1. Parametric Model. Assume Gaussian likelihood $P(Y|X) \sim N(\mu_X, \Sigma)$. Denote prior as $\pi(i)$. Use training data to estimate $\hat{\pi}, \hat{\Sigma}, \hat{\mu}_X$. Then Bayes classifier is linear y:

$$g(y) = \arg \max_{i\in\{1,2\}} \{\log \pi(i) - \frac{1}{2}(y-\mu_i)'\Sigma^{-1}(y-\mu_i)\}$$

$$= \arg \max_{i\in\{1,2\}} \{\log \pi(i) + \frac{1}{2}(2\mu_i'\Sigma^{-1}y - \mu_i'\Sigma^{-1}\mu_i)\}$$

$$\boxed{\text{LDA}: \hat{g}(y) = \arg \max_{i\in\{1,2\}} \{\log \hat{\pi}(i) + \frac{1}{2}(2\hat{\mu}_i'\Sigma^{-1}y - \hat{\mu}_i'\hat{\Sigma}^{-1}\hat{\mu}_i)\}}$$

LDA Decision threshold is straight line in y:

$$\log \hat{\pi}(1)+\frac{1}{2}(2\hat{\mu}_1'\Sigma^{-1}y-\hat{\mu}_1'\hat{\Sigma}^{-1}\hat{\mu}_1) = \log \hat{\pi}(2)+\frac{1}{2}(2\hat{\mu}_2'\Sigma^{-1}y-\hat{\mu}_2'\hat{\Sigma}^{-1}\hat{\mu}_2)$$

Example (cont): From data $\hat{\mu}_1 = [3.05, 6.38]'$; $\hat{\mu}_2 = [2.67, 4.73]'$, $\Sigma^{-1} = \begin{bmatrix} 5.745 & 0.791 \\ 0.791 & 0.701 \end{bmatrix}$, $\hat{\pi}(1) = 4/7, \hat{\pi}(2) = 3/7$.
LDA decision threshold is $3.49y_1 + 1.46y_2 = 17.78$ (see figure).

Method 2. Semi-Parameteric Model. Assume posterior

$$P(X = 1|Y) = \frac{\exp(\alpha + \beta' y)}{1 + \exp(\alpha + \beta' y)} \text{ (called a logistic model)}$$

where parameters α, β are estimated from training data. Then

$$P(X = 1|Y) > 1/2 \iff \exp(\alpha + \beta' y) > 1 \iff \alpha + \beta' y > 0$$

$$\boxed{\text{Semi-parametric classifier is } \hat{g}(y) = \begin{cases} 1 & \alpha + \beta' y > 0 \\ 2 & \text{otherwise} \end{cases}}$$

Logistic Regression with Supervised Learning

Linear regression model: $y_k = \psi_k'\theta + e_k$. Not useful when $y_k \in \{0,1\}$
Logistic regression model: $y_k \in \{0,1\}$ label generated probabilistically:

$$P(y_k = 1|\theta) = \sigma(\psi_k'\theta) = \frac{1}{1+\exp(-\psi_k'\theta)}$$

$\psi_k \in \mathbb{R}^n$ (feature) is known input vector at time k, $\theta \in \mathbb{R}^n$,
$\sigma(\cdot)$ = neuron: Sigmoidal (logistic function) with range $[0,1]$.

Aim. Given N training samples $(\psi_k, y_k), k = 1,\ldots,N$, compute maximum likelihood estimate θ^*. Assuming $\{y_k\}$ iid

$$\max_\theta L(\theta) = \log P(y_1,\ldots,y_N|\theta) = \sum_{k=1}^N \log P(y_k|\theta)$$

$$= \sum_{k=1}^N \log P(y_k = 1|\theta) I(y_k = 1) + \log P(y_k = 0|\theta) I(y_k = 0)$$

$$= \sum_{k=1}^N y_k \log \sigma(\psi_k'\theta) + (1-y_k)\log\big(1-\sigma(\psi_k'\theta)\big)$$

Off line gradient algorithm to compute MLE is:

$$\theta^{(I+1)} = \theta^{(I)} + \epsilon_I \nabla_\theta L(\theta)|_{\theta^{(I)}} = \theta^{(I)} + \epsilon_I \sum_{k=1}^N \psi_k \big(y_k - \sigma(\psi_k'\theta)\big)$$

Logistic stochastic gradient algorithm with step size $\epsilon_k = 1/k$ is:

$$\boxed{\theta_{k+1} = \theta_k + \epsilon_k \psi_k \big(y_k - \sigma(\psi_k'\theta_k)\big)}$$

Likelihood, Prior, Bayesian Inference

1. How to compute likelihood $p(y|x)$?
2. How to specify prior $p(x)$?
3. How to compute $p(x|y) = \frac{p(y|x)p(x)}{\int p(y|\zeta)p(\zeta)d\zeta}$?

1. Likelihood Formula to compute $p(y|x)$.

Suppose $Y = \phi(X) + W$, and W has cdf F_W and pdf f_W. Then $F_{Y|X}(y|x) = P(Y \le y|X = x) = P(\phi(X) + W \le y|X = x) = P(W \le y - \phi(x)) = F_W(y - \phi(x))$. Take derivative wrt y:

$$\boxed{f_{Y|X}(y|x) = \frac{dF}{dy} F_{Y|X}(y|x) = f_W(y - \phi(x))}$$

Example 1: $Y = X + W$ where $W \sim N(0, \sigma^2)$. Then

$$f_{Y|X}(y|x) = f_W(y - x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\frac{(y-x)^2}{\sigma^2}\right]$$

Example 2: $Y = AX + DW$ where $A, D \in \mathbb{R}^{n\times n}$ and D is positive definite. Show that $p(y|x) = \frac{1}{|D|} p_W(y - Ax)$

2. How to specify prior? Philosophical question; not based on observations

1. Single prior (classical)
2. Family of priors indexed by hyper parameter θ.
   (i) Hierarchical Bayes: $p(y|x)$, $p(x|\theta)$, $p(\theta)$.
   (ii) Empirical Bayes: $p(y|x)$, $p(x|\theta)$; estimate θ from data.
   (iii) Robust Bayes - choose non-informative prior which treats all parameters equally.

Two types of priors: Non-informative and informative.

Non-informative Priors. also called diffuse or flat prior.

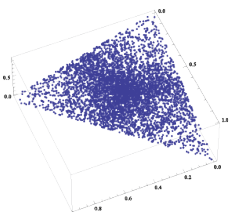
When nothing is known about x choose $p(x)$ as uniform pdf.

Example. Hierarchical Bayes Model and Dirichlet prior: used in Latent Dirichlet Allocation (LDA) in NLP.

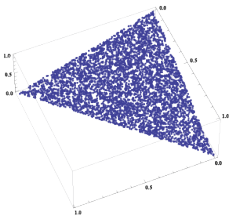
1. θ is an X-dim pmf chosen uniformly from all X-dim pmfs. $\theta \sim U(\Pi(X))$ where $\Pi(X) = \{\pi : \sum_{i=1}^{X} \pi(i) = 1, \pi(i) \geq 0\}$? $\Pi(X)$ is space of X-dim pmfs = unit $X-1$-dim simplex. (Example: $\theta = [0.1, 0.5, 0.4]'$.)
2. $x \sim \theta$. (Example. $x = 2$ with prob 0.5)
3. observation $y \sim p(y|x)$. (Example. $y = x +$ noise)

How to sample uniformly from space of X-dimensional pdfs? $\Pi(X) = \{\pi : \sum_{i=1}^{X} \pi(i) = 1, \pi(i) \geq 0\}$? $\Pi(X)$ is space of X-dim pmfs = unit $X-1$-dimensional simplex.
Ans: Dirichlet distribution.

Algorithm for sampling uniformly from $\Pi(X)$.
(a) Sample $\zeta(i) \sim e^{-\zeta}, \zeta \geq 0, i = 1, \ldots, X$ (unit expo density)
(b) Set $\theta(i) = \zeta(i) / \sum_j \zeta(j)$.



Normalized Uniform Distr.



Normalized Exp. Distr.

Note: Generating $\zeta(i) \sim U[0, 1]$ in step (a) does not work.

Jeffreys Prior (Advanced)

For discrete rv, discrete uniform prior is non-informative.
Monotone fn of disc uniform is disc uniform.
For continuous rv, uniform prior not invariant to transformation. No info about x implies no info about x^2; but uniform prior for x is not equivalent to uniform prior for x^2.
Jeffreys prior: pdf that is invariant to monotone transformation.
What is invariance of prior $p(x)$? If $z = h(x)$, then pdf of z is

$$q(z) = \frac{p(h^{-1}(z))}{|h'(h^{-1}(z))|}$$

Invariance to transformation $h(x)$ means pdf p satisfies

$$p(z) = q(z) \equiv p(z) = \frac{p(h^{-1}(z))}{|h'(h^{-1}(z))|} \implies \boxed{p(h(z)) = \frac{p(z)}{|h'(z)|}}$$

Fisher information: measures curvature of log-likelihood

$$I(x) = \mathbb{E}_y\{\left(\nabla_x \log p(y|x)\right)^2\} = -\mathbb{E}_y\{\nabla_x^2 \log p(y|x)\}$$

High curvature implies deeper valley: easier to estimate x.
Cramer Rao bound: For unbiased estimator $\hat{x}$, $\mathrm{Var}\{\hat{x}\} \geq 1/I(x)$.

Jeffreys prior: $\boxed{p(x) \propto \sqrt{I(x)}}$ Jeffreys prior is scale invariant: For monotone function $h(x)$, it can be shown that FIM satisfies

$$I(x) = I(h(x))(\nabla_x h(x))^2$$

So $p(x) \propto \sqrt{I(x)}$ satisfies $p(x) \propto \underbrace{\sqrt{I(h(x))}}_{p(h(x))} |\nabla_x h(x)|$.

Example 1. Gaussian obs with prior on mean; so $x = \mu$:

$$p(y|\mu) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2}\frac{(y-\mu)^2}{\sigma^2})$$

$I(x) = -\mathbb{E}_y\{\nabla^2_\mu \log p(y|\mu)\} = -\mathbb{E}_y\{\nabla^2_\mu \frac{(-y-\mu)^2}{2\sigma^2}\} = \frac{1}{\sigma^2}$.
Jeffreys prior for mean is $p(\mu) =$ constant indpt of μ, i.e. uniform prior. Improper prior since it cannot be a valid density.

Example 2. Gaussian obs with prior on standard deviation; so $x = \sigma$:

$$p(y|\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2}\frac{(y-\mu)^2}{\sigma^2})$$

Jeffreys prior for std deviation is $p(\sigma) = \sqrt{I(\sigma)} \propto \frac{1}{\sigma}$.

Example 3. Gaussian obs with prior on variance; so $x = \sigma^2$:
Jeffreys prior for variance is $p(\sigma^2) = \sqrt{I(\sigma^2)} \propto (\sigma^2)^{-3/2}$.

Improper Priors. Since $p(x|y) = \frac{p(y|x)p(x)}{\int p(y|\zeta)p(\zeta)d\zeta}$, it follows that $p(x)$ does not have to be a valid pdf for $p(x|y)$ to be a valid pdf.

Example: $p(x) =$ positive constant for $x \in \mathbb{R}$ is an improper prior. Results in proper posterior if $p(y|x)$ is valid pdf.

3. How to compute posterior distribution?

Bayes rule: $p(x|y) = \frac{p(y|x)p(x)}{\int p(y|\zeta)p(\zeta)d\zeta}$
In general, obtaining $p(x|y)$ in closed form is intractable due to denominator involving multidim integral.

Two ways:

1. Analytically: Conjugate Priors (if nice structure) gives closed form for posterior
2. Numerically: MCMC sampling from posterior for large scale problems - computational Bayesian (hot area in machine learning and signal processing)

Conjugate Priors

Bayes: posterior $\propto$ likelihood $\times$ prior

Suppose prior & likelihood chosen s.t. posterior has same pdf family as prior. Then prior & likelihood form *conjugate pair*.

Example 1. Gaussian prior and likelihood are conjugate. Prior $x \sim N(\mu, \sigma_w^2)$; likelihood $y \sim N(x, \sigma_v^2)$, then Gaussian posterior.

Notation. X-variate Gaussian with mean μ and cov P:

$$\boldsymbol{N}(x; \mu, P) = (2\pi)^{-X/2}|P|^{-1/2} \exp\left(-\frac{1}{2}(x-\mu)' P^{-1}(x-\mu)\right).$$

Theorem 5 (Swiss-Army-Knife for Gaussians)**.** *Consider Gaussian likelihood* $p(y|x) = \boldsymbol{N}(y; Cx, R)$ *and prior* $\boldsymbol{N}(x; \mu, P)$.

$$\begin{aligned} \textit{Then } \; p(y,x) &= \boldsymbol{N}(y; Cx, R)\, \boldsymbol{N}(x; \mu, P) \\ &= \boldsymbol{N}(y; C\mu, CPC' + R)\; \boldsymbol{N}(x; m, P - \bar{K}CP) \\ &\textit{where } \bar{K} = PC'(CPC' + R)^{-1}, \quad m = \mu + \bar{K}(y - C\mu). \end{aligned}$$

As a result, the following hold:

$$p(y) = \int_{\mathcal{X}} \boldsymbol{N}(y; Cx, R)\boldsymbol{N}(x; \mu, P)dx = \boldsymbol{N}(y; C\mu, CPC' + R) \qquad (6)$$

$$p(x|y) = \frac{\boldsymbol{N}(y; Cx, R)\boldsymbol{N}(x; \mu, P)}{\int_{\mathcal{X}} \boldsymbol{N}(y; Cx, R)\boldsymbol{N}(x; \mu, P)dx} = \underbrace{\boldsymbol{N}(x; \mu + \bar{K}(y - C\mu), P - \bar{K}CP)}_{\textit{posterior given } y}.$$

Main point: first Gaussian on RHS only involves y and not x. To marginalize product of Gaussians by integrating over x, we can take the first term on RHS outside integral and second term on RHS integrates to 1 – this yields (6).

There are several examples of conjugate priors.

1. Prior $x \sim N(\mu, \sigma_w^2)$; likelihood $y \sim N(x, \sigma_v^2)$ (above example).
2. Prior $\sigma^{-2} \sim \text{Gamma}(\alpha, \beta)$, likelihood $y \sim N(\mu, \sigma^2)$

See wikipedia for several additional examples.

Next: Computational Bayesian using MCMC. Key idea:

> *A cloud of samples from a distribution is equivalent to knowing the distribution*

Real life examples of offline Bayesian Inference:

1. Latent Dirichlet Allocation in NLP for topic modeling (in text mining, social media analysis, etc).
2. Optimal Search and Resource Allocation (dynamic spectrum allocation, defense applications)
3. Clinical Medical Trials (survival analysis for censored data)
4. Target localization (signal processing, defense)
5. Linear Discriminant Analysis (pattern recognition, face recognition)

Simulation Part 2: Markov Chain Monte-Carlo (MCMC)

Outline

1. Cont-state Markov Processes
2. Reversible Markov Chains
3. Metropolis Hasting Algorithm
4. MCMC for Bayesian inference
5. Gibbs Sampling for Bayesian inference

Aside: Continuous state Markov Chains

Consider stochastic difference equation:

$$x_{k+1} = \phi_k(x_k, w_k), \quad x_0 \sim \pi_0$$

State x_k lies in the state space $\mathcal{X} = \mathbb{R}^X$.
State noise $\{w_k\}$: iid sequence. Assume $\{w_k\}$, and x_0 are indpt.
Then x_k is a continuous state Markov process on $\mathbb{R}^X$:

$$\mathbb{P}(x_{n+1} \in S | x_1, x_2, \ldots, x_n) = \mathbb{P}(x_{n+1} \in S | x_n)$$

for any set $S \subseteq \mathbb{R}^X$.

Define *transition density* $P_{xy} = p(x_{k+1} = y | x_k = x)$: For $S \subseteq \mathcal{X}$,

$$\mathbb{P}(x_{k+1} \in S | x_k) = \int_S p(x_{k+1} = x | x_k)\, dx, \quad \int_{\mathcal{X}} p(x_{k+1} = x | x_k)\, dx = 1.$$

How to specify transition density? Use likelihood formula:
Suppose additive noise

$$x_{k+1} = A_k(x_k) + \Gamma_k(x_k) w_k, \quad x_0 \sim \pi_0$$

Assuming $\Gamma_k(x_k)$ is square invertible matrix,

$$\boxed{p(x_{k+1}|x_k) = |\Gamma_k^{-1}(x_k)| \;\; p_w\left(\Gamma_k^{-1}(x_k)\,[x_{k+1} - A_k(x_k)]\right)}$$

Example: If $x_{k+1} = a x_k + w_k$, $w_k \sim N(0,1)$, transition density

$$P_{x_k, x_{k+1}} = p(x_{k+1}|x_k) \propto \frac{1}{\sqrt{2\pi}} \exp\Big(-\frac{1}{2}(x_{k+1} - a x_k)^2\Big)$$

Markov Chain Monte-Carlo (MCMC)

For irreducible aperiodic Markov process, stationary distribution of the transition probability matrix (transition density) satisfies

Finite state Markov chain: $\pi_\infty(j) = \sum_i P_{ij}\pi_\infty(i) \iff \pi_\infty = P'\pi_\infty$

Cont. state Markov chain: $\pi_\infty(y) = \int P_{xy}\pi_\infty(x)dx$

---

1. We know how to simulate a Markov chain $\{x_k\}$ with transition matrix (density)
2. Sample path $\{x_k\}$ has stationary distribution π_∞.
   SLLN implies: (i) empirical cdf of x_k converges to cdf π_∞
   (ii) $\frac{1}{N}\sum_{k=1}^N h(x_k) \to \sum_i h(i)\pi_\infty(i)$ as $N \to \infty$.

MCMC Key idea:

1. Given a target distribution π_∞, find a transition prob P with stationary distribution π_∞.
2. Simulate Markov process $\{x_k\}$ with this transition prob P.

How to construct transition matrix P so that π_∞ is stationary distribution? Metropolis-Hastings algorithm constructs a *reversible* Markov chain

---

Why sample from a Markov chain rather than simulate i.i.d.?
(i) impossible to simulate i.i.d. samples from multidim cdf.
(ii) often π_∞ known up to a normalizing constant. Computing normalizing constant is intractable. e.g. Bayes rule

Reversible Markov Chains

A MC with transition prob P and stationary distribution π_∞ is *reversible* if it satisfies "balanced equation"

$$\boxed{P_{ij}\,\pi_\infty(i) = P_{ji}\,\pi_\infty(j) \quad \text{OR} \quad P_{xy}\pi_\infty(x) = P_{yx}\pi_\infty(y)}$$

Remarks: Recall $\pi_\infty = P'\pi_\infty$. (i) If a probability vector π_∞ satisfies balanced eq, then π_∞ must be stationary dist (Why?) (ii) Suppose $\{x_k\}$ is irreducible aperiodic Markov chain. Then

$$P(x_{n-1} = j | x_n = i, x_{n+1}, \ldots, x_{n+k}) = P(x_{n-1} = j | x_n = i) = \frac{\pi(j)P_{ji}}{\pi(i)}$$

So time-reversed process is Markov with transition probability

$$Q_{ij} = P(x_{n-1} = j | x_n = i) = \frac{\pi(j)P_{ji}}{\pi(i)}$$

In reversible Markov chain: $Q_{ij} = P_{ij}$, i.e, forward and backward transition probabilities are identical. So

$$P_{ij} = \frac{\pi(j)P_{ji}}{\pi(i)} \iff P_{ij}\pi(i) = P_{ji}\pi(j)$$

Running process forward or backward are statistically identical. (iii) All eigenvalues of P are real. Reason: P is *similar* to a symmetric matrix, i.e., $\exists T$s.t. $TPT^{-1} = S$ for symmetric S.

Proof: Define $\Pi_\infty \stackrel{\text{defn}}{=} \text{diag}(\pi_\infty(1), \cdots, \pi_\infty(X))$.

Reversibility: $\Pi_\infty P = P'\Pi_\infty \implies \Pi_\infty^{1/2} P \Pi_\infty^{-1/2} = (\Pi_\infty^{1/2} P \Pi_\infty^{-1/2})'$

If $TPT^{-1} = S$, clearly P and S have identical eigenvalues since $\det(\lambda I - TPT^{-1}) = \det(T^{-1}T)\det(\lambda I - TPT^{-1})$

Why reversible Markov chains? Given π_∞, much easier to construct reversible transition matrix that satisfies componentwise relationship, than to construct transition matrix that admits π_∞ as an eigenvector. That is,

$$\text{Reversible MC:} \quad P_{ij}\,\pi_\infty(i) = P_{ji}\,\pi_\infty(j)$$

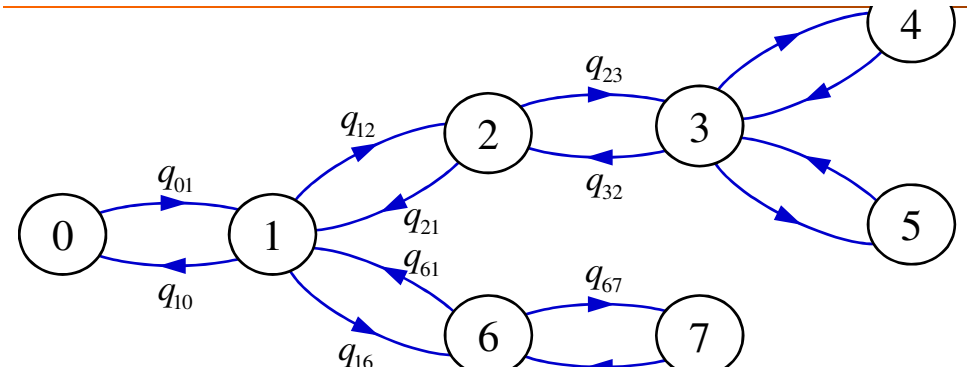
is simpler than $\pi_\infty(j) = \sum_i P_{ij}\pi_\infty(i)$ for general MC.

Also, optimizing the second largest eigenvalue modulus of a reversible transition matrix is a convex optimization problem.

How to tell if a Markov chain is reversible?

Result Sufficient condition: If the graph is a tree (contains no loops), then the Markov chain is reversible.

A Markov chain is a tree process if: (i) $P_{ij} > 0$ iff $P_{ji} > 0$. (ii) For every $i, j, i \neq j.$, there is a unique sequence of distinct states $i = i_0, i_1, i_2, \ldots, i_{n-1}, i_n = j$ such that $P_{i_k, i_{k+1}} > 0$.



(i) Any 2 state Markov chain is reversible.

(ii) Any tri-diagonal transition matrix is reversible.

(iii) Any symmetric transition matrix is reversible.

(iv) Not reversible if non-symmetric zero element.

Kolmogorov conditions. A Markov chain is reversible iff any path starting from state i and back to state i, has the same probability as going in the reverse direction;

$$P_{i,i_1} P_{i_1,i_2} \cdots P_{i_k,i} = P_{i,i_k} P_{i_k,i_{k-1}} \cdots P_{i_1,i} \qquad \forall k$$

Example (i): 3×3 transition matrix is reversible if

$$P_{12} P_{23} P_{31} = P_{13} P_{32} P_{21}$$

(ii) Any 2 state Markov chain is reversible.

Result. A transition matrix is not reversible if it has a non-symmetric zero element.

Outline

1. Cont-state Markov Processes
2. Reversible Markov Chains
3. Metropolis Hasting Algorithm
4. MCMC for Bayesian inference
5. Gibbs Sampling for Bayesian inference

Metropolis Hastings Algorithm

Aim: Simulate from *target distribution* π_∞ by constructing a reversible Markov chain that has π_∞ as stationary distribution.

Metropolis-Hastings (discrete rv)

- Choose any irreducible transition matrix Q *(proposal distribution)*.
- Define acceptance probability

$$\alpha_{ij} = \min\left(1, \frac{\pi_\infty(j)Q_{ji}}{\pi_\infty(i)Q_{ij}}\right)$$

- Given state $x_k = i$ at time k:
  1. Simulate next state of the Markov chain $j \sim Q_{ij}$.
  2. Generate $u \sim U[0,1]$.
  3. If $u < \alpha_{ij}$ then set $x_{k+1} = j$ (accept); otherwise set $x_{k+1} = x_k = i$ (reject).

Metropolis-Hastings (continuous rv)

- Choose any transition kernel $q(y|x) > 0$ for all $x, y \in \mathcal{X}$. $q(y|x)$ is called *proposal distribution*.
- Define

$$\alpha_{xy} = \min\left(1, \frac{\pi_\infty(y)q(x|y)}{\pi_\infty(x)q(y|x)}\right)$$

- Given state $x_k = x$ at time k:
  1. Simulate next state of the Markov chain $y \sim q_{y|x}$.
  2. Generate $u \sim U[0,1]$.
  3. If $u < \alpha_{xy}$ then set $x_{k+1} = y$ (accept); otherwise set $x_{k+1} = x_k = x$ (reject).

Theorem: $\{x_k\}$ generated by Metropolis Hastings is a reversible Markov chain with stationary distribution π_∞.

Remarks:

1. Step 3 accepts $x_{k+1} = j$ with probability α_{ij} and remains in the state $x_k = i$ with probability $1 - \alpha_{ij}$.
2. Target distribution π_∞ only needs to be known up to a normalizing constant, since the acceptance probabilities of step 3 depend on the ratio $\pi_\infty(j)/\pi_\infty(i)$. This is crucially important in Bayesian inference (sampling from posterior).
3. If proposal distribution Q is chosen symmetric $Q_{ij} = Q_{ji}$

$$\alpha_{ij} = \min(1, \frac{\pi_\infty(j)}{\pi_\infty(i)})$$

   This is called random-walk Metropolis algorithm.
4. *Proof that Metropolis Hastings algorithm works*:
   Markov chain $\{x_k\}$ has transition probabilities

$$P_{ij} = \alpha_{ij} Q_{ij}, \quad j \neq i \qquad \text{where } \alpha_{ij} = \min\left(1, \frac{\pi_\infty(j) Q_{ji}}{\pi_\infty(i) Q_{ij}}\right) \tag{7}$$

   We prove x_k is a reversible MC, i.e., $P_{ij}\pi_\infty(i) = P_{ji}\pi_\infty(j)$.

$$P_{ij}\pi_\infty(i) = \alpha_{ij} Q_{ij} \pi_\infty(i) = \min(Q_{ij}\pi_\infty(i), Q_{ji}\pi_\infty(j))$$

   Suppose $\alpha_{ij} < 1$. Then (7) implies
   (a) $Q_{ji}\pi_\infty(j) < Q_{ij}\pi_\infty(i)$ so $P_{ij}\pi_\infty(i) = Q_{ji}\pi_\infty(j)$.
   (b) $\alpha_{ji} = 1$. So $P_{ji}\pi_\infty(j) = \alpha_{ji} Q_{ji}\pi_\infty(j) = Q_{ji}\pi_\infty(j)$.
   Therefore $P_{ij}\pi_\infty(i) = P_{ji}\pi_\infty(j)$.
   Hence x_k is reversible MC with stationary distribution π_∞.

Example 1. MCMC for n-dimensional numerical integration

$$\mathbb{E}\{h(X)\} = \int_{\mathbb{R}^n} h(x)p(x)dx \leftarrow \frac{1}{N}\sum_{k=1}^{N} h(x_k)$$

Suppose $p(x)$ is difficult to simulate from - so use MCMC.
Soln. Choose symmetric $q(y|x) = q(x|y)$ e.g. multidim Gaussian

$$q(y|x) \propto \exp\big(-\frac{1}{2}(x-y)'\Sigma^{-1}(x-y)\big)$$

This is the random walk MH algorithm: $\alpha(x,y) = \min(\frac{p(y)}{p(x)}, 1)$.

Markov process $\{x_k\}$ generated by MH has stationary dist $p(x)$.

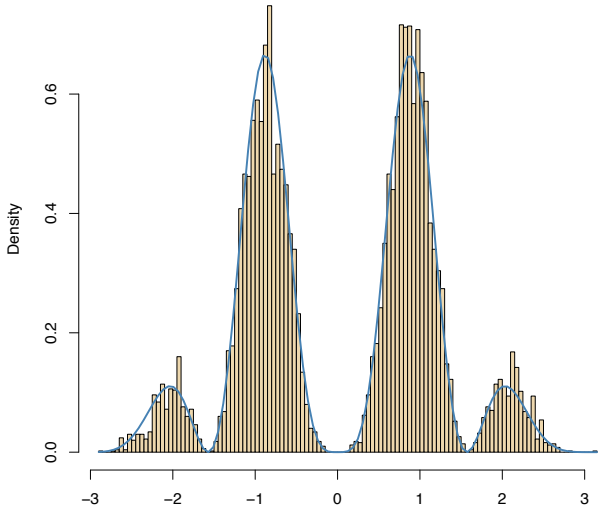
Example 2. Let $\psi(x) = N(0,1)$. Simulate from

$$\pi(x) \propto \sin^2(x) \times \sin^2(2x) \times \psi(x)$$

Proposal: $q(y|x) = U(x-\alpha, x+\alpha) = \frac{1}{2\alpha} I(x-\alpha \le y \le x+\alpha)$.
Clearly $q(y|x)$ is symmetric: $q(x|y) = q(y|x)$.

$$q(x|y) = \frac{1}{2\alpha} I(y-\alpha \le x \le y+\alpha) = \frac{1}{2\alpha} I(x-\alpha \le y \le x+\alpha)$$

So $\alpha_{xy} = \min(1, \pi(y)/\pi(x))$.



Metropolis Hastings for 10^4 iterations $\alpha = 1$ starting at $x_0 = 3.14$.

MCMC for Computational Bayesian

How to numerically implement Bayes rule?

$$p(x|y) = \frac{p(y|x)p(x)}{\int_{\mathbb{R}^X} p(y|\zeta)p(\zeta)d\zeta}$$

Denominator: high dimensional integral; difficult to evaluate.

Key idea: Given observation y, use Metropolis Hastings to simulate Markov chain x_k with target (stationary) distribution

$$\pi(x) \propto p(y|x)p(x), \quad k = 1, 2, \ldots$$

For convenience, use a random walk Metropolis Hastings:

1. Simulate x_k with symmetric transition $q(\bar{x}|x) = q(x|\bar{x})$
2. acceptance probability $\alpha_{x,\bar{x}} = \min\left(1, \frac{p(y|\bar{x})p(\bar{x})}{p(y|x)p(x)}\right)$

A cloud of samples $\{x_k\}$ is equivalent to knowing pdf or pmf.

We can then use MCMC simulated samples $\{x_k\}$ to:

1. estimate posterior cdf or pdf. Empirical pdf of posterior is

$$\hat{p}_n(x|y) = \frac{1}{n}\sum_{k=1}^{n} I(x_k \in [x - \Delta, x + \Delta))$$

   Matlab: Given $x = [x_1, \ldots, x_n]$, use `hist(x, nbins, 1)`
2. estimate conditional mean: by law of large numbers

$$\frac{1}{n}\sum_{k=1}^{n} x_k \to \mathbb{E}\{x|y\} = \int x p(x|y) dx$$

Remark. Marginalization: Given samples $\alpha_k, \beta_k \sim p(\alpha, \beta)$ then samples $\alpha_k \sim p(\alpha)$.

Example 1- Binomial with improper prior.

We observe number of wins S_n in n indpt trials.
Prior probability of winning in each trial is rv $X \sim p(X)$.

1. Estimate posterior distribution $p(X|S_n)$.
2. Estimate conditional mean $\mathbb{E}\{X|S_n\}$.

Model. Prior: $\boxed{p(X) \propto 2\cos^2(4\pi X)}$ (improper prior), $x \in [0,1]$.
$P(Y_i = 1 \text{ (win)}) = X$ for $i = 1, 2, \ldots, n$ (Bernoulli rv).
Observation: number of wins $S_n = \sum_{i=1}^n Y_i$.

So likelihood is: $p(S_n = s|X) = \binom{n}{s} X^s (1-X)^{n-s}$ (binomial)

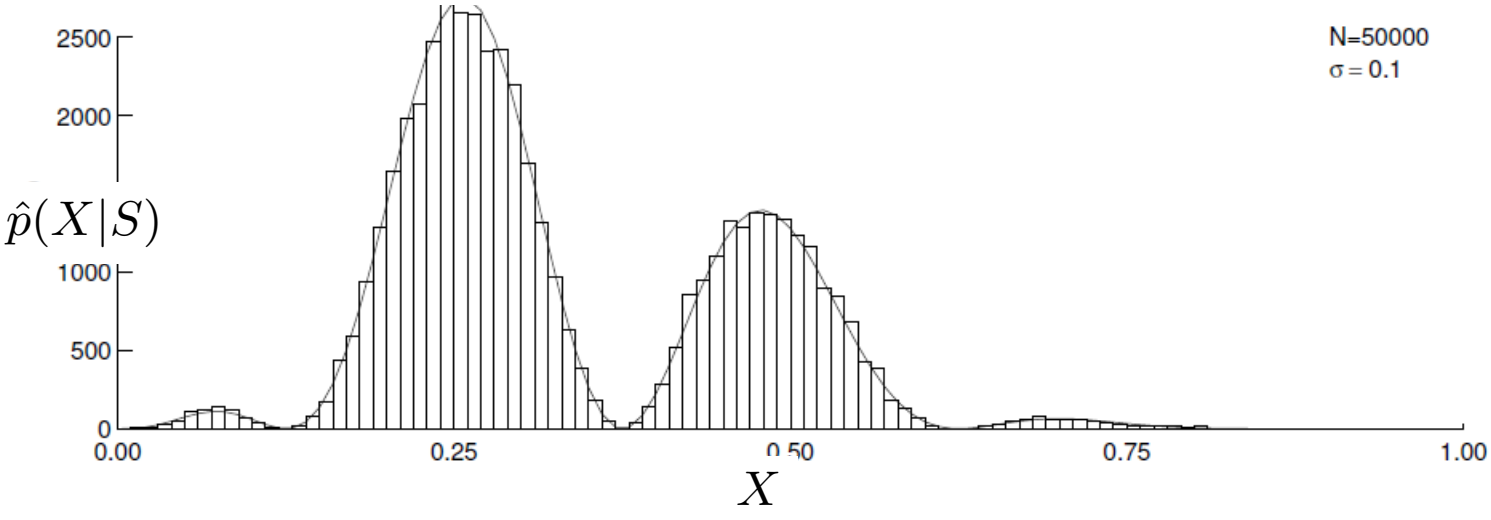
Procedure: Given S_n, use MH to simulate samples from posterior

$$p(X|S_n) \propto X^{S_n}(1-X)^{n-S_n} \cos^2(4\pi X)$$

Soln: Choose symmetric proposal in Metropolis Hastings:

$$\text{simulate } x_k \sim q(\bar{X}|X) \propto \exp\left(-\frac{1}{2\sigma^2}(X-\bar{X})^2\right)$$

$$\alpha_{X,\bar{X}} = \min\{1, \frac{\bar{X}^{S_n}(1-\bar{X})^{n-S_n}\cos^2(4\pi\bar{X})}{X^{S_n}(1-X)^{n-S_n}\cos^2(4\pi X)}$$



Example 2 - Covariation for Hierarchical Bayes Model

Given two data streams of observations $x_{1:N}$ and $y_{1:N}$.

Aim: Compute posterior estimate of correlation ρ between data streams, i.e. compute $\mathbb{E}\{\rho|x_{1:N}, y_{1:N}\}$. Assume

$$x_i, y_i|\rho \sim N(\mu, \Sigma(\rho)), \quad \mu \in \mathbb{R}^2, \Sigma(\rho) = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$$

for $i = 1, \ldots, N$. So likelihood

$$p(x_{1:N}, y_{1:N}|\rho) = \prod_{i=1}^{N} \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}[x_i^2 - 2\rho x_i y_i + y_i^2]\right)$$

Assume prior is $p(\rho) \propto (1-\rho^2)^{-3/2}$.

So from Bayes rule, the posterior

$$p(\rho|x_{1:N}, y_{1:N}) \propto \text{ prior } \times \text{ likelihood}$$

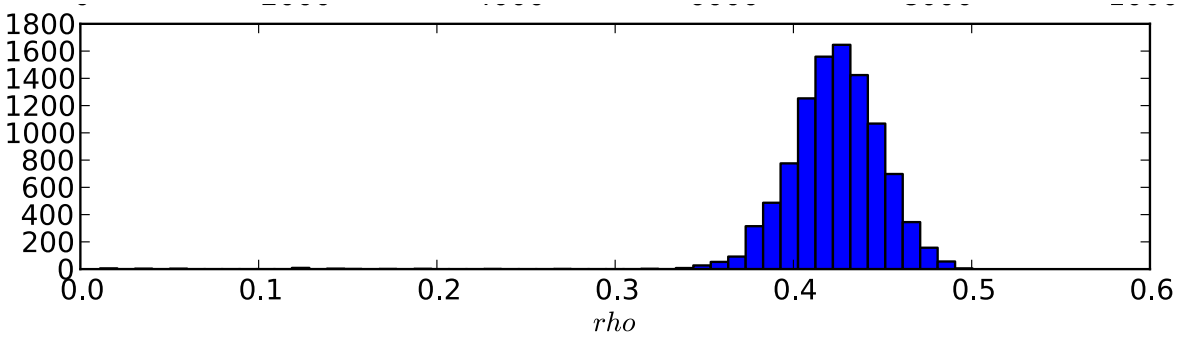
We use MH to sample from posterior $p(\rho|x_{1:N}, y_{1:N})$ and then estimate conditional mean $\mathbb{E}\{\rho|x_{1:N}, y_{1:N}\}$.

Choose a symmetric kernel $q(\rho_{k+1}|\rho_k) = U(\rho_k - 0.07, \rho_k + 0.07)$.

Then acceptance probability is

$$\alpha_{\rho_{k-1}, \rho_k} = \min\{1, \frac{p(\rho_k|x_{1:N}, y_{1:N})}{p(\rho_{k-1}|x_{1:N}, y_{1:N})}\}$$

Simulation for $N = 10000$ time points. Posterior distribution histogram based on posterior samples:



Estimate of $\mathbb{E}\{\rho|x_{1:10000}, y_{1:10000}\} = 0.42$, std dev $= 0.03$.

Example 3. Hierarchical Bayesian Models.

$$\Theta \sim p(\theta) \quad \text{(hyper-parameter)}$$
$$X|\Theta \sim p(x|\theta)$$
$$Y|X \sim p(y|x)$$

Aim: Given y, sample from posterior $p(x|y)$.

Step 1. Use Metropolis Hastings to generate samples $(\theta, x)_i$, $i = 1, 2, \ldots$ from

$$p(\theta, x|y) \propto p(y|x)\, p(x|\theta)\, p(\theta)$$

(i) Simulate (θ_k, x_k) with symmetric transition density $q(\bar{x}, \bar{\theta}|x, \theta) = q(x, \theta|\bar{x}, \bar{\theta})$

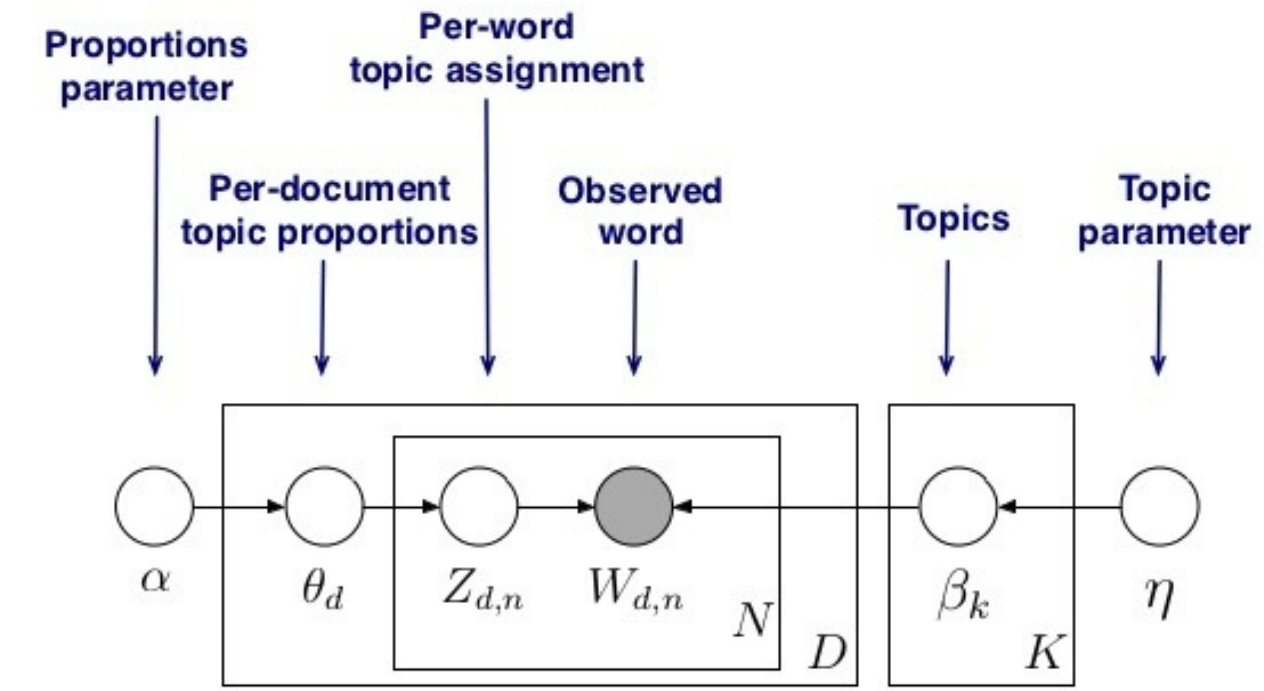
(ii) acceptance probability

$$\alpha_{x,\theta,\bar{x},\bar{\theta}} = \min\left(1, \frac{p(y|\bar{x})p(\bar{x}|\bar{\theta})p(\bar{\theta})}{p(y|x)p(x|\theta)p(\theta)}\right)$$

Step 2. Then samples x_i, $i = 1, \ldots$ are from posterior $p(x|y)$.

Example: **Latent Dirichlet Allocation** (LDA) for *topic modeling* in natural language processing: understand what document describes and retrieve documents that describe topic.

LDA as a graphical model



$$\prod_{i=1}^{K} p(\beta_i \mid \eta) \prod_{d=1}^{D} p(\theta_d \mid \alpha) \left(\prod_{n=1}^{N} p(z_{d,n} \mid \theta_d) p(w_{d,n} \mid \beta_{1:K}, z_{d,n}) \right)$$

See the following famous paper

`http://ai.stanford.edu/~ang/papers/jair03-lda.pdf`

Gibbs Sampling

Special case of the Metropolis-Hastings algorithm.
Used in 2 related contexts:
(i) To sample from multivariate $p(x_1, x_2, \ldots, x_L)$ when it is easy to sample from conditional univariate distributions,

$$p(x_1|x_2, \ldots, x_L), \quad p(x_2|x_1, x_3, \ldots, x_L), \ldots$$

(ii) (Data augmentation) Difficult to sample from $p(x)$ but there exists latent variable y such that $p(x, y)$ can be Gibbs sampled via $p(x|y)$ and $p(y|x)$ Then samples of x yields marginal $p(x)$.
Example: censored data models in survival analysis

Gibbs Sampling Algorithm from $\pi_\infty(x) = p(x_1, \ldots, x_L)$
Each iteration n has L stages.

- Given samples $x_1^{(n)}, x_2^{(n)}, \ldots, x_L^{(n)}$ from iteration n.
  At $(n+1)$th iteration generate
  1. $x_1^{(n+1)} \sim p(x_1|x_2^{(n)}, x_3^{(n)}, \ldots, x_L^{(n)})$
  2. $x_2^{(n+1)} \sim p(x_2|x_1^{(n)}, x_3^{(n)}, \ldots, x_L^{(n)})$
  $\vdots$
  L. $x_L^{(n+1)} \sim p(x_L|x_1^{(n)}, x_2^{(n)}, \ldots, x_{L-1}^{(n)})$

Remarks: Can also use $x_2^{(n+1)} \sim p(x_2|x_1^{(n+1)}, x_3^{(n)}, \ldots, x_L^{(n)})$, etc

1. Markov chain $(x_1^{(n)}, x_2^{(n)}, \ldots, x_L^{(n)})$, $n = 1, 2, \ldots$ has empirical pdf that converges to $\pi_\infty(x) = p(x_1, \ldots, x_L)$.
2. Gibbs sampling is a special case of Metropolis Hastings.
3. Order of updating L coordinates can be chosen randomly.

Result. Gibbs sampling = composition of L Metropolis-Hastings algorithms each with acceptance probability $\alpha = 1$.
Denote $x = (x_1, x_2, x_3)$, $y = (x_1, \bar{x}_2, x_3)$, $\pi_\infty(x) = p(x)$, $\pi_\infty(y) = p(y)$.

$$Q_{xy} = \frac{1}{3}p(y|x) = \frac{1}{3}p(\bar{x}_2|x_1, x_3) = \frac{1}{3}\frac{p(x_1, \bar{x}_2, x_3)}{p(x_1, x_3)} = \frac{1}{3}\frac{p(y)}{p(x_1, x_3)}$$

$$Q_{yx} = \frac{1}{3}\frac{p(x)}{p(x_1, x_3)}$$

Having specified proposal Q, acceptance probabilities in Metropolis-Hastings algorithm are

$$\alpha = \frac{\pi_\infty(y)Q_{yx}}{\pi_\infty(x)Q_{xy}} = \frac{p(y)Q_{yx}}{p(x)Q_{xy}} = 1 \quad \text{i.e. no rejections}$$

Example 1: Bivariate Normal. see movie at
`http://gorayni.blogspot.com/2013/08/gibbs-sampling.html`

Aim: Simulate $(x, y) \sim \boldsymbol{N}(0, \Sigma)$, $\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$, $|\rho| < 1$.

Verify using Swiss-army knife formula that

$$p(x|y) = \boldsymbol{N}(\rho y, 1 - \rho^2), \qquad p(y|x) = \boldsymbol{N}(\rho x, 1 - \rho^2)$$

Gibbs Sampler Simulate for $n = 1, 2, \ldots$

1. $x^{(n+1)} \sim p(x|y^{(n)})$,
2. $y^{(n+1)} \sim p(y|x^{(n+1)})$

This converges to stationary distribution $\boldsymbol{N}(0, \Sigma)$.

See paper: Sampling Truncated Normal, Beta, and Gamma Densities Paul Damien & Stephen G Walker, Journal of Computational and Graphical Statistics, 2001.

Example 2. Bayesian Inference for Censored Data Models

Motivation. (i) Survival analysis: How to determine effect of drug on average life-span; in a fixed-time study? People who survive fixed-time study length yield no info on life span.
Let y_k denote date of death.

Model.
$$z_k = \begin{cases} y_k & \text{if } y_k \le c \\ * & \text{otherwise} \end{cases}, \quad y_k \sim p(y_k|x), \quad x \sim p(x)$$

Assume observations are $z_{1:m} = y_{1:m}$ and $z_{m+1:n} = *$.
$*$ is a *right censored* observation (data right of c is censored)
Aim: How to sample from posterior $p(x|z_1, \ldots, z_n)$? Evaluating posterior in closed form is impossible in general.

Motivation. (ii) Quality Control: Test a product over c years to estimate time to failure. If product still functions after c years, then censored observation.

Classical MCMC: $\pi(x) \propto p(z_{1:n}|x)\, p(x)$. Often specifying/evaluating likelihood $p(z_{1:n}|x)$ is difficult.
Trick. Use data augmentation.

1. Augment posterior with fictitious observations $y_{m+1:n}$; then use Gibbs sampling on $p(x, y_{m+1:n}|z_{1:n})$.
2. Then marginal $p(x|z_{1:n})$ is obtained from the empirical pdf of simulated samples $x^{(i)}, i = 1, 2, \ldots$.

III.2. Optimal Bayesian Filtering

Aim: The key question answered here is:
Given a stochastic signal observed in noise, how does one construct an optimal estimator of the signal?
The key results will be covered using elementary concepts in probability and stochastic processes. Optimal Filters are used in telecommunication systems, radar tracking systems, speech processing, machine learning, robotics

- Stochastic State Space Models & Optimal Predictors
- Kalman Filter, HMM filter
- Briefly describe sequential MCMC based particle filters.
- Describe how these filters can be used for ML parameter estimation (if time permits)

Stochastic Difference Equation to Transition Density

$$\begin{aligned} x_{k+1} &= A_k(x_k) + \Gamma_k(x_k) w_k, \quad x_0 \sim \pi_0 \\ y_k &= C_k(x_k) + D_k(x_k) v_k. \end{aligned} \tag{8}$$

Assume $\Gamma_k(x_k)$ and $D_k(x_k)$ are square invertible matrices.
Then transition density and observation likelihood are:

$$\begin{aligned} p(x_{k+1}|x_k) &= |\Gamma_k^{-1}(x_k)|\; p_w\left(\Gamma_k^{-1}(x_k)\,[x_{k+1} - A_k(x_k)]\right) \\ p(y_k|x_k) &= |D_k^{-1}(x_k)|\; p_v\left(D_k^{-1}(x_k)\,[y_k - C_k(x_k)]\right). \end{aligned}$$

where $|\cdot|$ denotes determinant.

Example: Moving Autonomous Vehicle Linear Gaussian state space model

$$x_{k+1} = Ax_k + Bu_k + v_k$$

$x_k \stackrel{\text{defn}}{=} [r_x[k], \dot{r}_x[k], r_y[k], \dot{r}_y[k]]'$: state vector at time kT
T is the sampling interval.
$r_x(k)$ is x-coordinate position, $\dot{r}_x(k)$ is velocity in x direction.

$$A = \begin{pmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{pmatrix}$$

Optimal Prediction: Chapman Kolmogorov Equation

Aim: How to optimally predict future state given current state probability?
Given Markov process with transition density $p(x_{k+1}|x_k)$ and initial condition π_0, compute state pdf $\pi_k(x) = p(x_k = x)$ at time k.

We call π_k the *predicted* density.

Chapman Kolmogorov: From total probability rule

$$\boxed{\pi_k(x) = \int_{\mathcal{X}} p(x_k = x|x_{k-1})\, \pi_{k-1}(x_{k-1}) dx_{k-1}, \quad \text{initialized by } \pi_0.}$$

Therefore predicted state and covariance at time k are

$$\boxed{\begin{aligned} \hat{x}_k = \mathbb{E}\{x_k\} &= \int_{\mathcal{X}} x\, \pi_k(x) dx, \\ \text{cov}(x_k) = \mathbb{E}\{(x_k - \hat{x}_k)(x_k - \hat{x}_k)'\} &= \mathbb{E}\{x_k x_k'\} - \hat{x}_k \hat{x}_k'. \end{aligned}}$$

Predicted mean is $\hat{x}_k$ is optimal in the minimum mean square error sense:

$$\mathbb{E}\{(x_k - \hat{x}_k)^2\} \leq \mathbb{E}\{(x_k - \phi(\pi_0))^2\}.$$

Hence called "optimal predictor".

Ex 1. Linear Gaussian State Space Model

Notation:

$$\boldsymbol{N}(\zeta;\mu,\Sigma) = (2\,\pi)^{-l/2}\,|\Sigma|^{-1/2}\exp\left[-\frac{1}{2}(\zeta-\mu)'\Sigma^{-1}(\zeta-\mu)\right].$$

Sometimes shorter notation $\boldsymbol{N}(\mu,\Sigma)$ will be used.

The linear Gaussian state space model

$$\boxed{\begin{aligned} x_{k+1} &= A_k x_k + w_k, \quad x_0 \sim \pi_0 = \boldsymbol{N}(\hat{x}_0,\Sigma_0),\ w_k \sim \boldsymbol{N}(0,Q_k) \\ y_k &= C_k x_k + v_k, \quad v_k \sim \boldsymbol{N}(0,R_k). \end{aligned}}$$

$x_k \in \mathcal{X} = \mathbb{R}^X$, $y_k \in \mathcal{Y} = \mathbb{R}^Y$, $A_k \in \mathbb{R}^{X\times X}$, $C_k \in \mathbb{R}^{Y\times X}$.

Transition density form:

$$p(x_{k+1}|x_k) = p_w\left(x_{k+1} - A_k(x_k)\right) = \boldsymbol{N}(x_{k+1}; A_k x_k, Q_k)$$
$$p(y_k|x_k) = p_v\left(y_k - C_k(x_k)\right) = \boldsymbol{N}(y_k; C_k x_k, R_k).$$

Optimal Predictor Using the Chapman Kolmogorov equation $\pi_{k+1} = \boldsymbol{N}(\hat{x}_{k+1},\Sigma_{k+1})$ where

$$\boxed{\begin{aligned} \hat{x}_{k+1} &= \mathbb{E}\{x_{k+1}\} = A_k\hat{x}_k \\ \Sigma_{k+1} &= \mathrm{cov}\{x_{k+1}\} = A_k\Sigma_k A_k' + Q_k. \end{aligned}}$$

Covariance update is called *Lyapunov equation.*

Same mean and covariance recursions also hold for non-gaussian case

Ex 2. Hidden Markov Model

Markov chain measured via a noisy observation process.

(i) *Markov chain*: $\{x_k\} \in \mathcal{X} = \{1, 2, \ldots, X\}$.
Transition probability matrix P with elements

$$P_{ij} = \mathbb{P}(x_{k+1} = j | x_k = i), \quad 0 \le P_{ij} \le 1, \quad \sum_{j=1}^{X} P_{ij} = 1 \equiv P\mathbf{1} = \mathbf{1}$$

Initial: $\mathbb{P}(x_0 = i) = \pi_0(i)$, $i = 1, \ldots, X$ with $\sum_{i=1}^{X} \pi_0(i) = 1$.
State level vector (physical states) $C = [C(1), C(2), \ldots, C(X)]'$.

(ii) *Noisy Observations*: Define *observation probability* density

$$y_k \sim B_{xy} = p(y_k = y | x_k = x), \quad x \in \mathcal{X}, y \in \mathcal{Y}.$$

Note $\sum_y B_{xy} = 1$ or $\int B_{xy} dy = 1$

Example 1: Additive noise HMM. $\mathcal{Y} = \mathbb{R}$,

$$y_k = C(x_k) + D(x_k)\, v_k, \quad v_k \sim p_v$$

Then using likelihood formula

$$B_{xy} = p(y_k = y | x_k = x) = \frac{1}{D(x)}\, p_v\Big(\frac{y - C(x)}{D(x)}\Big)$$

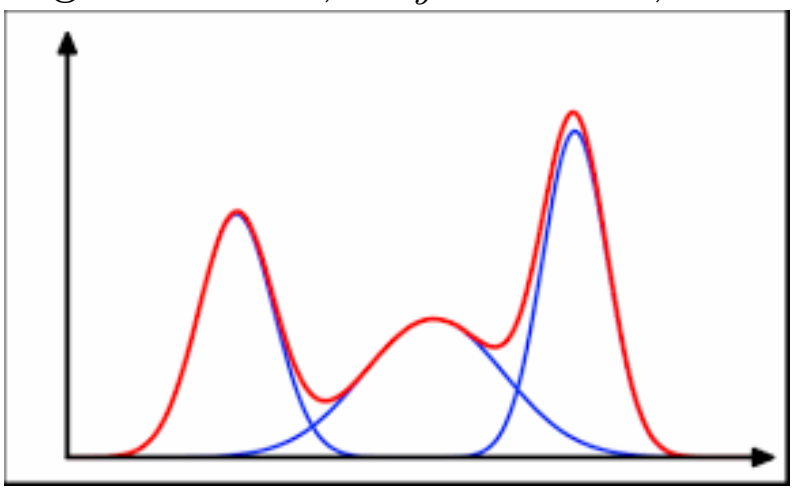
Example 2: Finite Observation Space HMM. $\mathcal{Y} = \{1, \ldots, Y\}$.
Then $B_{xy} = \mathbb{P}(y_k = y | x_k = x)$ are called "symbol" probabilities.

$$B_{X \times Y} = \begin{bmatrix} 0.8 & 0.2 & 0 \\ 0.1 & 0.3 & 0.6 \end{bmatrix}$$

HMM= *dynamic mixture model.*

If $P = \text{iid}$, then HMM specializes to classical mixture model.

E.g. if $P = \text{iid}$, B_{xy} Gaussian, then Gaussian mixture model.



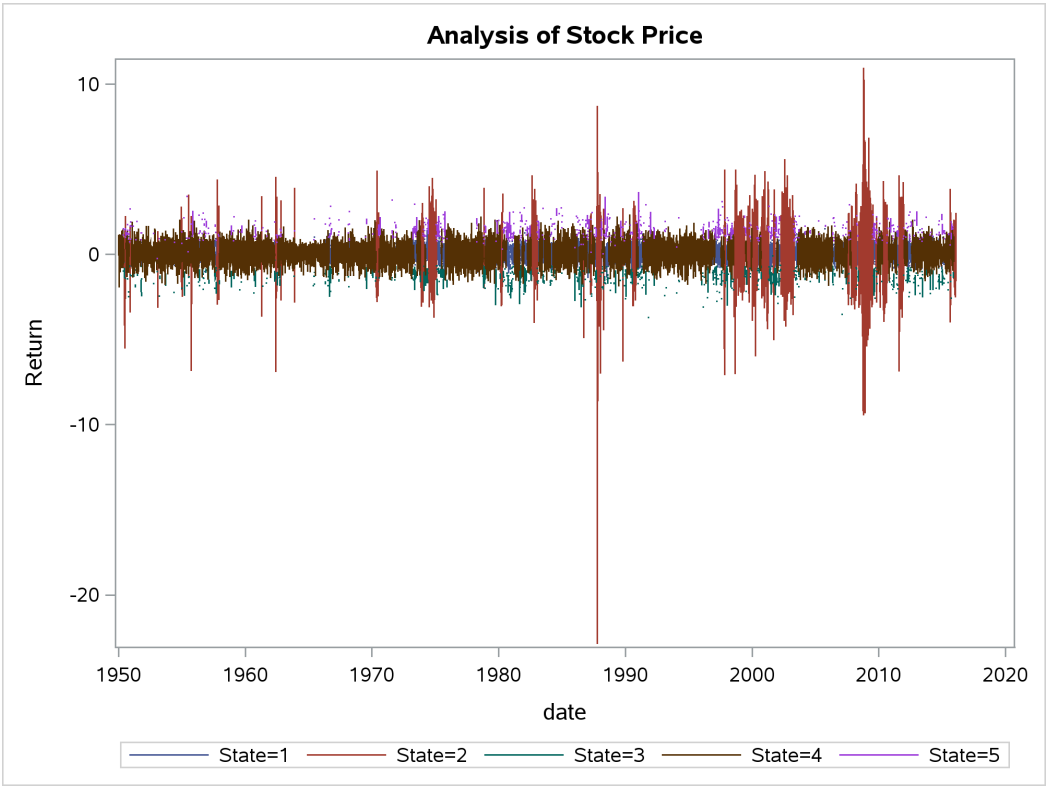
$X = 3$ and

$$p(y) = \sum_{i=1}^{X} \pi(i) B_{iy}$$

where $B_{iy} = p(y|x = i)$ is Gaussian.

HMM (dynamic mixture): $p(y) = \sum_{i=1}^{X} \pi_k(i) B_{iy}$.

HMMs and mixture models used in: Topic analysis in documents (LDA), speech recognition, neurobiology, bioinformatics, financial models, econometrics, wireless comms, target tracking, handwriting recognition, image segmentation



Optimal Predictor Define state probability vector at time k

$$\pi_k = \begin{bmatrix} \mathbb{P}(x_k = 1) & \dots & \mathbb{P}(x_k = X) \end{bmatrix}'. \tag{9}$$

Then the Chapman-Kolmogorov equation reads

$$\pi_{k+1} = P'\pi_k \quad \text{initialized by } \pi_0. \tag{10}$$

So (10) is the optimal predictor for an HMM. Then

$$\mathbb{E}\{C(x_{k+1})\} = \sum_{i=1}^{X} C(i)\pi_{k+1}(i) = C'\pi_{k+1}.$$

Limiting Distribution *Limiting distribution* is

$$\lim_{k\to\infty} \pi_k = \lim_{k\to\infty} {P'}^k \pi_0.$$

This limiting distribution may not exist. For example if

$$P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \pi_0 = \begin{bmatrix} \pi_0(1) \\ \pi_0(2) \end{bmatrix}, \quad \text{then } \pi_k = \begin{cases} \begin{bmatrix} \pi_0(2) & \pi_0(1) \end{bmatrix}' & k \text{ odd} \\ \begin{bmatrix} \pi_0(1) & \pi_0(2) \end{bmatrix}' & k \text{ even} \end{cases}$$

and so $\lim_{k\to\infty} \pi_k$ does not exist unless $\pi_0(1) = \pi_0(2) = 1/2$.

Stationary Distribution X-dimensional vector π_∞

$$\pi_\infty = P'\pi_\infty, \quad \mathbf{1}'\pi_\infty = 1$$

So π_∞ is normalized right eigenvector of P' corresponding to the unit eigenvalue. Equivalently, choosing $\pi_0 = \pi_\infty$ implies $\pi_k = \pi_\infty$ for all k. The stationary distribution is also called the *invariant, equilibrium or steady-state distribution.*

Limiting distributions are a subset of stationary distributions. For example, given P in (181), $\pi_\infty = \begin{bmatrix}0.5 & 0.5\end{bmatrix}'$ is a stationary distribution but there is no limiting distribution.

Theorem 6 (Perron-Frobenius)**.** *Consider a finite-state Markov chain with regular transition matrix P. Then:*

1. *The eigenvalue 1 has algebraic & geometric multiplicity of one.*
2. *All remaining eigenvalues of P have modulus strictly smaller than 1.*
3. *The eigenvector of P' corresponding to eigenvalue of 1 can be chosen with non-negative elements.*
4. *$P^k = \mathbf{1}\pi'_\infty + O(|\lambda_2|^k)$ where λ_2 is the second largest eigenvalue modulus.*
5. *The limiting distribution and stationary distribution coincide.*

Statement 4 says if the transition matrix P is regular, state probability vector π_k forgets initial condition geometrically fast.

$$\pi_k = {P'}^k \pi_0 = \pi_\infty \mathbf{1}'\pi_0 + O(|\lambda_2|^k)\pi_0 = \pi_\infty + O(|\lambda_2|^k)\pi_0.$$

So k-step ahead predictor of a Markov chain forgets initial condition geometrically fast in terms of the second largest eigenvalue modulus, $|\lambda_2|$.
Equivalently, π_k converges geometrically fast to π_∞.

Ex 3. Jump Markov Linear Systems

$$\begin{aligned} z_{k+1} &= A(r_{k+1})\, z_k + \Gamma(r_{k+1})\, w_{k+1} + f(r_{k+1})\, u_{k+1} \\ y_k &= C(r_k)\, z_k + D(r_k)\, v_k + g(r_k) u_k. \end{aligned}$$

where r_k is finite state Markov chain, u_k is known input.

1. JMLS (Switched Markov Linear System) widely used stochastic state space models to model maneuvering moving objects, deconvolution, finance.
2. Models hybrid systems: continuous state real life system interaction with finite state automata.
3. If r_k is one state, then classical linear system.
   If $A = 1$, $\Gamma = f = 0$, then HMM.

Augmented state process $x_k = (z_k, r_k)$ is a Markov process in state space $\mathcal{X} = \mathbb{R}^z \times \{1, \ldots, X\}$.

The transition density of the JMLS state x_k is

$$\begin{aligned} & p\left(x_{k+1} = (\bar{z}, j) | x_k = (z, i)\right) \\ &= p(r_{k+1} = j | r_k = i) p\left(z_{k+1} = \bar{z} | r_{k+1} = j, z_k = z\right) \\ &= P_{ij}\, |\Gamma^{-1}(j)|\, p_w\left(\Gamma^{-1}(j)\left[\bar{z} - A(j)z - f(j)u_{k+1}\right]\right). \end{aligned}$$

The observation likelihood is evaluated as

$$p(y_k | x_k = (z, i)) = |D^{-1}(i)|\, p_v\left(D^{-1}(i)\left[y_k - C(i)z - g(i)u_k\right]\right).$$

Outline

- **Part I. Bayesian Inference and MCMC** (6 classes)
  1. Things to know (1)
  2. Simulation-1 (1)
  3. Bayes Inference of Random Variables (1)
  4. Simulation-2: MCMC (3)
- **Part II. Bayesian Filtering and Graphical Models** (12 classes)
  1. Stochastic State Space Models & Optimal Predictors (2)
  2. Bayesian Filtering and Smoothing (4)
  3. Graphical Models and Social Learning (2)
- **Part III Maximum Likelihood and EM** (4 hours)
  MLE for Hidden Markov Models
- **Part IV. Bayesian Decision Making** (9 classes)
  1. Markov Decision Processes
     (a) Stochastic Dynamic Programming
  2. Partially Observed Markov Decision Processes
     (a) POMDP Model and applications
     (b) Stochastic Dynamic Programming
     (c) Bayesian Stopping Time Problems

Bayesian Filtering

Wiener Filter: Nobert Wiener 1940s:
Model $Y = S + W$, S is signal W is noise.

$$\min_F \mathbb{E}\|S - FY\|^2$$

Widely used in LMMSE detection.

Kalman Filter: (1960s) Model S and N in time domain (state space models). The Kalman filter is probably the single most used algorithm in signal processing.

Hidden Markov Filter: Developed by statisticians (L. Baum, T. Petrie) in 1960s
Significant application in Electrical Engg in 1990s in speech recognition, channel equalization, tracking, etc

Sequential Markov Chain Monte Carlo Methods:
Particle filters – randomized (simulation based) algorithms – applications in target tracking – late 1990s.

Stochastic Filtering theory studies optimal filtering. Also called recursive Bayesian estimation.

In continuous-time stochastic filtering theory involves stochastic calculus – widely used in mathematical finance. Not covered here.

Aim of Optimal Filtering

Given a partially observed stochastic dynamical system

$$x_{k+1} = A_k(x_k) + \Gamma_k(x_k)w_k, \quad x_0 \sim \pi_0(\cdot)$$
$$y_k = C_k(x_k) + D_k(x_k)v_k,$$

$\{v_k\}$ and $\{w_k\}$ are iid. In transition density form

$$p(x_{k+1}|x_k) \propto p_w\left(\Gamma_k^{-1}(x_k)\,[x_{k+1} - A_k(x_k)]\right)$$
$$p(y_k|x_k) \propto p_v\left(D_k^{-1}(x_k)\,[y_k - C_k(x_k)]\right).$$

Assume known model and parameters.
Aim: Compute state estimate

$$\boxed{\hat{x}_k = \mathbb{E}\{x_k|y_1,\ldots,y_k\} = \int_{\mathbb{R}^X} x\, p(x_k = x|y_1,\ldots,y_k)dx}, k = 1,2,\ldots$$

Aim: Compute the min-variance state estimate $\hat{x}_{l|k}$ given the sequence of observations $y_{1:k} = y_1,\ldots,y_k$.
As shown $\hat{x}_{l|k} = \mathbb{E}\{x_l|y_{1:k}\}$; and is unbiased estimator.

There are 3 problems of interest:

- **Filtering**. $k = l$ (real time): compute $p(x_k|y_{1:k})$.
- **Prediction**. $l > k$ (real time): compute $p(x_{k+\Delta}|y_{1:k})$.
- **Smoothing**: $l < k$. (off-line): compute $p(x_{k-\Delta}|y_{1:k})$.

We focus here on filtering.
Predictor = filter with non-information observations: $p(y|x)$ indpt of x.
Smoother: forward and backward filter discussed later

Remark. Bayesian state estimation vs Adaptive filtering (RLS, LMS).

Filtering Recursion

Model : $p(x_{k+1}|x_k) \propto p_w\left(\Gamma_k^{-1}(x_k)\,[x_{k+1} - A_k(x_k)]\right)$,

$p(y_k|x_k) \propto p_v\left(D_k^{-1}(x_k)\,[y_k - C_k(x_k)]\right)$, $\pi_0(x) = p(x_0 = x)$.

Aim: For $k = 1, 2\ldots$, recursively compute filtered density

$$\pi_k(x) = p(x_k = x|y_{1:k})$$

Then conditional mean (optimal) state estimate is

$$\hat{x}_k = \mathbb{E}\{x_k|y_{1:k}\} = \int_{\mathcal{X}} x_k\, p(x_k|y_{1:k}) dx_k, \quad k = 1, 2\ldots$$

Main Result: Starting with $\pi_0(x)$, for $k = 0, 1, \ldots$

$$\pi_{k+1}(x_{k+1}) = \frac{p(y_{k+1}|x_{k+1}) \int_{\mathcal{X}} p(x_{k+1}|x_k)\, \pi_k(x_k)\, dx_k}{\int_{\mathcal{X}} p(y_{k+1}|x_{k+1}) \int_{\mathcal{X}} p(x_{k+1}|x_k)\, \pi_k(x_k)\, dx_k dx_{k+1}}$$

$$\hat{x}_{k+1} = \mathbb{E}\{x_{k+1}|y_{1:k+1}\} = \int_{\mathcal{X}} x\, \pi_{k+1}(x) dx$$

Remarks:

1. With only 2 exceptions (Kalman & HMM filter) posteriors $\{\pi_k\}$ are not finite dimensional computable.
2. Predictor is special case of filter with $p(y|x)$ indpt of x:

$$\pi_{k+1}(x_{k+1}) = \int_{\mathcal{X}} p(x_{k+1}|x_k)\, \pi_k(x_k)\, dx_k \qquad \text{(CK eqn)}$$

Prediction & Measurement Form of Optimal Filter Starting with $\pi_0(x)$, for $k = 0, 1, \ldots$

$$\boxed{\begin{aligned}\pi_{k+1|k}(x_{k+1}) &\stackrel{\text{defn}}{=} p(x_{k+1}|y_{1:k}) = \int_{\mathcal{X}} p(x_{k+1}|x_k)\,\pi_k(x_k)\,dx_k \\ \pi_{k+1}(x_{k+1}) &= \frac{p(y_{k+1}|x_{k+1})\pi_{k+1|k}(x_{k+1})}{\int_{\mathcal{X}} p(y_{k+1}|x_{k+1})\pi_{k+1|k}(x_{k+1})dx_{k+1}}\end{aligned}}$$

1. First step is Chapman Kolomogorov eqn for predictor
2. Second step is Bayesian update

Un-normalized Filter Update

Recall $\pi_k(x) = p(x_k = x|y_{1:k})$ is conditional density
It is often convenient to compute un-normalized density:

$$q_k(x) = p(x_k = x, y_{1:k}).$$

$$\text{Clearly} \quad \pi_k(x) = \frac{q_k(x)}{\int_{\mathcal{X}} q_k(x)dx}$$

Filtering recursion for un-normalized density is:
Start with $q_0(x) = \pi_0(x)$. Then for $k = 0, 1, \ldots$.

$$\boxed{q_{k+1}(x) = p(y_{k+1}|x_{k+1} = x) \int_{\mathcal{X}} p(x_{k+1} = x|x_k)q_k(x_k)dx_k.}$$

$$\hat{x}_{k+1} = \mathbb{E}\{x_{k+1}|y_{1:k+1}\} = \frac{\int_{\mathcal{X}} x\, q_{k+1}(x)dx}{\int_{\mathcal{X}} q_{k+1}(x)dx}.$$

Example. Toy Kalman Filter

Consider scalar linear Gaussian model

$$\begin{aligned} x_{k+1} &= x_k + w_k, \quad w_k \sim N(0,1); \quad \pi_0(x) \sim \boldsymbol{N}(\hat{x}_0, \Sigma_0). \\ y_k &= x_k + v_k, \quad v_k \sim N(0,1) \end{aligned}$$

$$\text{Then } p(y_k|x_k) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(y_k - x_k)^2\right)$$

$$p(x_{k+1}|x_k) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(x_{k+1} - x_k)^2\right), \ \pi_0(x) \sim N(\hat{x}_0, \Sigma_0).$$

Filtering recursion for $q_k(x) = p(x, y_{1:k})$ is: $q_0(x) \sim \boldsymbol{N}(\hat{x}_0, \Sigma_0)$

$$\begin{aligned} q_{k+1}(x_{k+1}) &= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(y_{k+1} - x_{k+1})^2\right) \\ &\quad \times \int_{\mathbb{R}} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(x_{k+1} - x_k)^2\right) q_k(x_k) dx_k \\ \pi_{k+1}(x_{k+1}) &= \frac{q_{k+1}(x_{k+1})}{\int_{\mathbb{R}} q_{k+1}(z) dz} \end{aligned}$$

How to solve for q_{k+1}? Suppose $q_k \sim \boldsymbol{N}(\hat{x}_k, \Sigma_k)$.

1. Since $q_k(x)$ is Gaussian, and convolution of Gaussians is Gaussian, the integral is Gaussian.
2. The integral is a Gaussian prior, the likelihood is Gaussian; since Gaussians are conjugate priors, $q_{k+1}(x)$ is Gaussian.

Key point. $q_k(x) \sim \boldsymbol{N}(\hat{x}_k, \Sigma_k) \implies q_{k+1}(x) \sim \boldsymbol{N}(\hat{x}_{k+1}, \Sigma_{k+1})$

Kalman filter: $\hat{x}_{k+1} = \phi_1(\hat{x}_k, \Sigma_k)$, $\Sigma_{k+1} = \phi_2(\Sigma_k)$.

Optimal Filter. Example 1. Kalman Filter

Model: Linear Gaussian State Space

$$x_{k+1} = A_k x_k + f_k u_k + w_k, \quad x_0 \sim \pi_0$$
$$y_k = C_k x_k + g_k u_k + v_k.$$

$w_k \sim \boldsymbol{N}(0, Q_k)$, $v_k \sim \boldsymbol{N}(0, R_k)$, $\pi_0 \sim \boldsymbol{N}(\hat{x}_0, \Sigma_0)$.

Kalman filter equations: (for conditional mean and covariance)

$$\hat{x}_{k+1|k} = A_k \hat{x}_k + f_k u_k, \quad y_{k+1|k} = C_{k+1} \hat{x}_{k+1|k} + g_{k+1} u_{k+1}$$
$$\Sigma_{k+1|k} = A_k \Sigma_k A_k' + Q_k$$
$$S_{k+1} = C_{k+1} \Sigma_{k+1|k} C_{k+1}' + R_{k+1}$$
$$\hat{x}_{k+1} = \hat{x}_{k+1|k} + \Sigma_{k+1|k} C_{k+1}' S_{k+1}^{-1} (y_{k+1} - y_{k+1|k})$$
$$\Sigma_{k+1} = \Sigma_{k+1|k} - \Sigma_{k+1|k} C_{k+1}' S_{k+1}^{-1} C_{k+1} \Sigma_{k+1|k}$$

Note that the posteriors are Gaussian:

$$p(x_k|y_{1:k}) = \boldsymbol{N}(\hat{x}_k, \Sigma_k),$$

where $\hat{x}_k = \mathbb{E}\{x_k|y_{1:k}\}, \quad \Sigma_k = \mathbb{E}\{(\hat{x}_k - x_k)(\hat{x}_k - x_k)'\}$

Also, $p(x_{k+1}|y_{1:k}) = \boldsymbol{N}(\hat{x}_{k+1|k}, \Sigma_{k+1|k})$ where

$$\hat{x}_{k+1|k} = \mathbb{E}\{x_{k+1}|y_{1:k}\},$$
$$\Sigma_{k+1|k} = \mathbb{E}\{(\hat{x}_{k+1} - x_{k+1})(\hat{x}_{k+1} - x_{k+1})'\}$$

Derivation of KF follows directly from Chapman Kolmogorov eqn and Swiss-army knife formula.

Prediction form of Kalman filter equations:

$$\hat{x}_{k+1|k} = (A_k - K_k C_k)\hat{x}_{k|k-1} + K_k\,(y_k - g_k u_k) + f_k u_k,$$
$$K_k = A_k \Sigma_{k|k-1} C_k' (C_k \Sigma_{k|k-1} C_k' + R_k)^{-1}$$
$$\Sigma_{k+1|k} = A_k \left(\Sigma_{k|k-1} - \Sigma_{k|k-1} C_k' (C_k \Sigma_{k|k-1} C_k' + R_k)^{-1} C_k \Sigma_{k|k-1}\right) A_k' + Q_k$$

K_k is called the Kalman filter gain.

Update for covariance $\Sigma_{k+1|k}$ is called Riccati equation. If $R_k \to \infty$ becomes Liapunov equation for optimal linear predictor.

KF vs RLS. If $R = I$, $Q = 0$, $g = f = 0$, KF predictor form yields RLS.

---

1. Kalman filter is optimal filter for linear Gaussian model - it yields the conditional mean estimate.

2. In general filtering recursion, we can only compute *conditional* covariance of the state estimate (conditioned on $y_{1:k}$)

$$\mathrm{cov}(x_k | y_{1:k}) = \int x^2 \pi_k(x) dx - \left(\int x \pi_k(x) dx\right)^2$$

Evaluating unconditional covariance requires integration over all possible observation trajectories is impossible.
Remarkably Kalman covariance Σ_k is unconditional:

$$\Sigma_k = \mathrm{cov}(x_k) = \int \left[\int x^2 p(x_k|y_{1:k}) - \left(\int x p(x_k|y_{1:k})\right)^2\right] p(y_{1:k}) dy_{1:k}$$

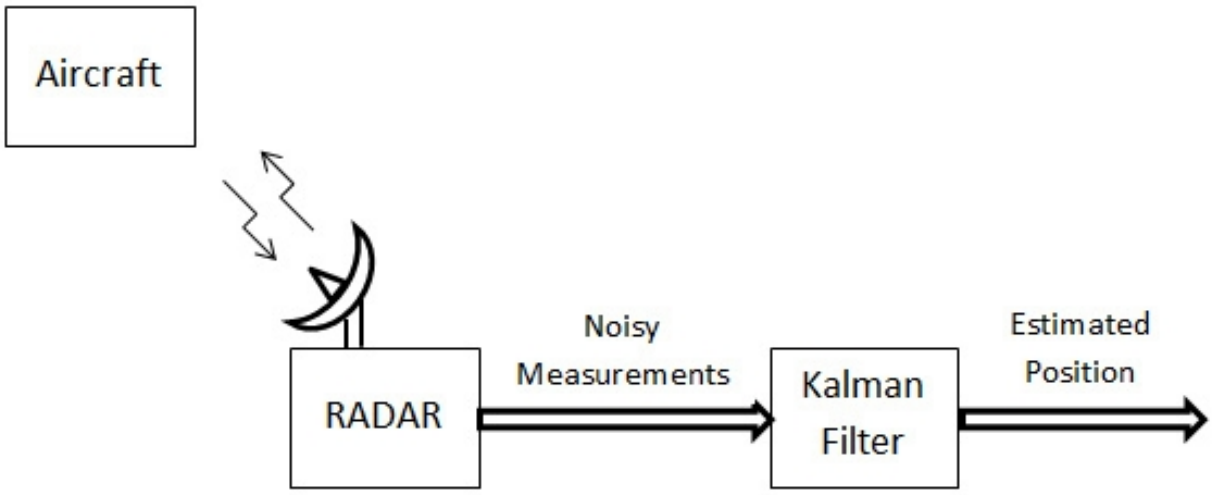
So KF automatically computes quality of state estimate.

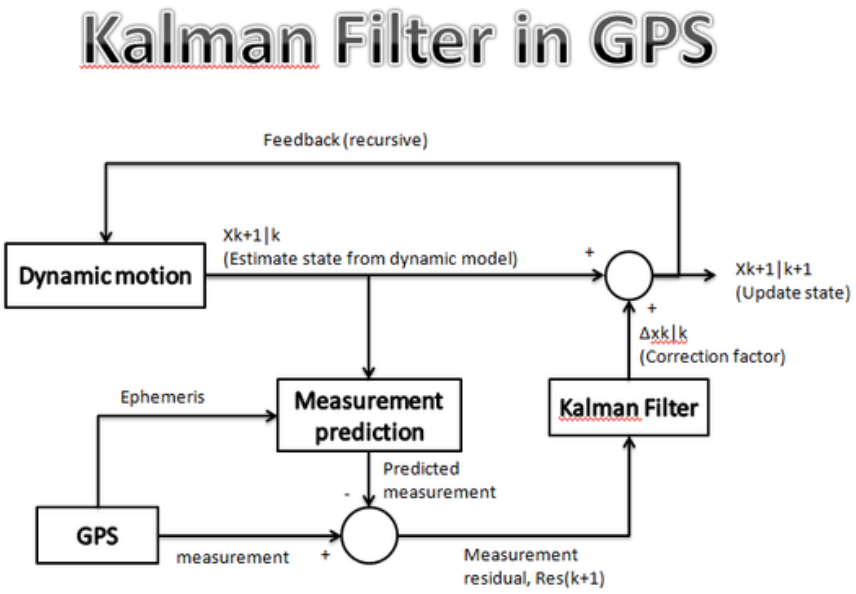
Kalman Predictor

$$\hat{x}_{k+i+1|k} = A_{k+i}\hat{x}_{k+i}, \quad \Sigma_{k+i+1|k} = A_{k+i}\Sigma_{k+i|k}A_{k+i}' + Q_{k+i}.$$

Predictor covariance update is called Liapunov equation

Examples of Kalman filters





`https://plus.maths.org/content/understanding-unseen` How NASA used Kalman filter for spacecraft navigation

Properties of the Kalman Filter

- Kalman Filter is linear, discrete-time, finite dimensional system with 2 sufficient statistics: $\hat{x}_k$, Σ_k.
- Covariance Σ_k can be precomputed since it is independent of the data. (unconditional covariance).
- Stability of KF is related to stability of

$$\lambda_{k+1} = (A_k - K_k C_k)\lambda_k$$

- Steady State Kalman Filter. If A, B, C, Q and R are time-invariant, then under stability conditions, K_k and Σ_k converge to a constant. Widely used in industrial control.
- In non-Gaussian noise, Kalman filter is linear minimum variance estimator: Amongst the class of linear estimators, Kalman filter is the minimum variance estimator.
- Above derivation is algebraic. Another method is basd on projection theorem (Hilbert space approach to linear functionals).
- Derivation of Kalman filter follows straightforwardly from Theorem 5

For details on Kalman filters see "Optimal Filtering" by B.D.O Anderson and J.B. Moore, Prentice Hall, 1979.

`https://www.mathworks.com/videos/`

`understanding-kalman-filters-part-1-why-use-kalman-filters--148581`

`html`

Optimal Filter Example 2. HMM Filter

Recall HMM is (P, B, π_0).

$$P_{ij} = \mathbb{P}(x_{k+1} = j | x_k = i), \quad B_{xy} = p(y_k = y | x_k = x)$$

HMM Filter: Since $\mathcal{X} = \{1, \ldots, X\}$, posterior π_k is X-dimensional pmf:

$$\pi_k(i) = \mathbb{P}(x_k = i | y_{1:k}), \quad i = 1, \ldots, X$$

$$\boxed{\pi_{k+1}(j) = \frac{p(y_{k+1}|x_{k+1} = j) \sum_{i=1}^X P_{ij} \pi_k(i)}{\sum_{l=1}^X p(y_{k+1}|x_{k+1} = l) \sum_{i=1}^X P_{il} \pi_k(i)}} \quad j = 1, \ldots, X,$$

In matrix-vector notation:

$$B_{y_k} = \text{diag}\begin{bmatrix} p(y_k|x_k = 1) & \cdots & p(y_k|x_k = X) \end{bmatrix}.$$

$$\pi_k = \begin{bmatrix} \pi_k(1) & \cdots & \pi_k(X) \end{bmatrix}'$$

$$\boxed{\pi_{k+1} = \frac{B_{y_{k+1}} P' \pi_k}{\mathbf{1}' B_{y_{k+1}} P' \pi_k}}$$

Compute the conditional mean estimate of $C' x_{k+1}$ as

$$C'(\hat{x}_{k+1}) = \mathbb{E}\{C(x_{k+1}) | y_{1:k+1}\} = C' \pi_{k+1}.$$

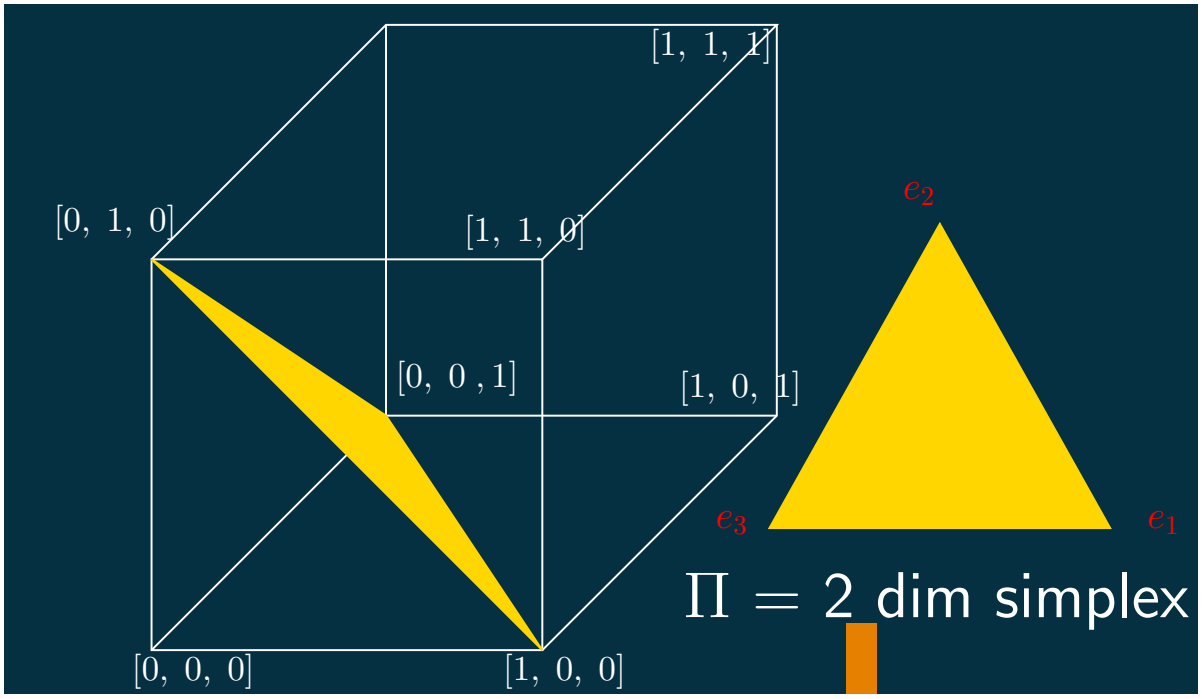
HMM filter requires $O(X^2)$ multiplications at each time instant.

Note: If $P = I$, then HMM is called mixture-model. HMM filter specializes to recursive Bayesian estimation of rv

Belief Space: Geometric interpretation

$$\Pi(X) \stackrel{\text{defn}}{=} \left\{ \pi \in \mathbb{R}^X : \mathbf{1}'\pi = 1, \quad 0 \le \pi(i) \le 1 \text{ for all } i \in \mathcal{X} \right\}$$

Unit vectors $e_1, e_2, \ldots, e_X$ are vertices of this simplex Π.



Un-normalized HMM filter and Forward algorithm

$$q_{k+1} = p(x_{k+1}, y_{1:k+1}) = B_{y_{k+1}} P' q_k.$$

$$\hat{x}_{k+1} = \mathbb{E}\{x_{k+1} | y_{1:k+1}\} = \frac{q_{k+1}}{\mathbf{1}' q_{k+1}}.$$

Called *forward* algorithm.

Scaling: underflow problem remedied by scaling all the elements of q_k by any arbitrary positive number. Since $\hat{x}_k$ involves the ratio of q_k with $\mathbf{1}'q_k$, this scaling factor cancels out in $\hat{x}_k$.

Fixed interval HMM Smoother

Aim: Given data $y_{1:N} = (y_1, \ldots, y_N)$, compute smoothed pmf

$$\pi_{k|N}(i) = P(x_k = i | y_{1:N}), \quad \text{for } k = 1, \ldots, N$$

Forward variable: $q_k(i) = P(x_k = i, y_{1:k})$. Backward variable:

$$\beta_{k|N}(i) = p(y_{k+1:N} | x_k = i), \ \beta_{k|N} = \Big[\beta_{k|N}(1), \ldots, \beta_{k|N}(X)\Big]'.$$

$$\implies p(x_k = i, y_{1:N}) = p(x_k = i, y_{1:k}) p(y_{k+1:N} | x_k = i) = q_k(i)\beta_{k|N}(i).$$

$$\implies \pi_{k|N}(i) = \frac{\frac{q_k(i)}{\mathbf{1}'q_k}\beta_{k|N}(i)}{\sum_{j=1}^{X} \frac{q_k(j)}{\mathbf{1}'q_k}\beta_{k|N}(j)} = \frac{\pi_k(i)\beta_{k|N}(i)}{\sum_{j=1}^{X} \pi_k(j)\beta_{k|N}(j)}$$

Backward vector $\beta_{k|N}$ is computed via backward recursion:

$$\boxed{\beta_{k|N} = PB_{y_{k+1}}\beta_{k+1|N}, \quad k = N-1, \ldots, 1, \quad \beta_{N|N} = \mathbf{1}_X}$$

Summary: Smoothed estimate $\pi_{k|N}(i) = p(x_k = i | y_{1:N})$, $i = 1, \ldots, X$ computed via *forward backward algorithm*:

$$\pi_k = \frac{B_{y_k} P' \pi_{k-1}}{\mathbf{1}' B_{y_k} P' \pi_{k-1}} \quad k = 1, 2, \ldots, N, \quad \text{(forward filter)}$$

$$\beta_{k|N} = PB_{y_{k+1}}\beta_{k+1|N}, \quad k = N-1, \ldots, 1, \ \beta_{N|N} = \mathbf{1} \quad \text{(backward)}$$

$$\pi_{k|N}(i) = \mathbb{P}(x_k = i | y_{1:N}) = \frac{\pi_k(i)\beta_{k|N}(i)}{\sum_{j=1}^{X} \pi_k(j)\beta_{k|N}(j)} \quad \text{(smoother)}$$

Scaling: Must scale $\beta_{k|N}$ during the backward recursion to avoid numerical underflow. Scale factor cancels out in $\pi_{k|N}$.

Remarks about forward-backward algorithm:

1. Derivation of backward recursion:

$$\begin{aligned}\beta_{k|N}(i) &= p(y_{k+1:N}|x_k = i) = p(y_{k+1}, y_{k+2:N}|x_k = i) \\ &= \sum_{j=1}^{X} p(y_{k+2:N}|x_{k+1} = j, y_{k+1}, x_k = i)p(x_{k+1} = j, y_{k+1}|x_k = i) \\ &= \sum_{j=1}^{X} \beta_{k+1|N}(j)\, B_{j,y_{k+1}}\, P_{ij}\end{aligned}$$

2. Initialization of $\beta_{N|N}$: At time N, foward backward algorithm yields:

$$\pi_{N|N}(i) = \mathbb{P}(x_N = i|y_{1:N}) = \frac{\pi_N(i)\beta_{N|N}(i)}{\sum_{l=1}^{m} \pi_N(l)\beta_{N|N}(l)}$$

But $\pi_N(i) = \pi_{N|N}(i)$. Therefore $\beta_{N|N}(i) = 1$ for all i.

3. Memory and Computational Cost: Forward backward algorithm needs

1. $O(XN)$ memory to store the X vectors π_k and $\beta_{k|N}$, $k = 1, \ldots, N$.
2. $O(X^2N)$ multiplications.

Smoothers for Functionals of State

Aim: Compute smoothed estimate of functional

$$\mathbb{E}\{\sum_{k=1}^{N} \phi(x_k)|y_{1:N}\}$$

Why? Need this in ML estimation algorithms for HMMs.

Ex1: $D_i(N)$ denote duration time in state i until time N. Then

$$\mathbb{E}\{D_i(N)|y_{1:N}\} = \mathbb{E}\{\sum_{k=1}^{N} I(x_k = i)|y_{1:N}\}$$

$$= \sum_{k=1}^{N} P(x_k = i|y_{1:N}) = \sum_{k=1}^{N} \pi_{k|N}(i)$$

Ex2: $J_{ij}(N)$: number of jumps from i to j until time N. Then

$$\mathbb{E}\{J_{ij}(N)|y_{1:N}\} = \mathbb{E}\{\sum_{k=1}^{N} I(x_{k-1} = i, x_k = j)|y_{1:N}\}$$

$$= \sum_{k=1}^{N} P(x_{k-1} = i, x_k = j|y_{1:N}) = \sum_{k=1}^{N} \frac{\pi_k(i) P_{ij} B_{jy_{k+1}} \beta_{k+1|N}(j)}{\sum_{l=1}^{X} \pi_k(l) P_{lj} B_{jy_{k+1}} \beta_{k+1|N}(j)}$$

Markov Modulated Auto-regressive Time Series

Used in econometric modelling and maneuvering targets.

Example: Let x_k be finite state Markov chain. Consider

$$y_k + a_1(x_k)\, y_{k-1} + \cdots + a_d(x_k)\, y_{k-d} = \Gamma(x_k) w_k,$$

Identical to HMM filter with observation density

$$B_{x,y_k} = p_w\left(\Gamma^{-1}(x)(y_k + a_1(x)\, y_{k-1} + \cdots + a_d(x)\, y_{k-d})\right).$$

Viterbi Algorithm for HMMs (FYI)

Recall HMM smoother computes conditional mean estimate

$$\mathbb{E}\{x_k|y_{1:N}\} = \operatorname*{argmin}_g \mathbb{E}\{g(y_{1:N}) - x_k\}^2$$

Given HMM obs sequence $y_{1:N} = (y_1, \ldots, y_N)$, Viterbi algorithm computes Maximum likelihood sequence estimate (MLSE):

$$\boxed{\hat{x}_{1:N} = \arg\max_{x_{1:N}} p(y_{1:N}, x_{1:N})}, \quad x_{1:N} = (x_1, \ldots, x_N).$$

$$\hat{x}_{1:N} = \arg\max_{x_{1:N}} p(y_{1:N}, x_{1:N}) = \arg\max_{x_{1:N}} \prod_{k=1}^{N} p(y_k|x_k)p(x_k|x_{k-1})p(x_1)$$

$$= \arg\max_{x_{1:N}} \sum_{k=1}^{N} \log p(y_k|x_k) + \log p(x_k|x_{k-1}) + \log p(x_1)$$

Viterbi algorithm computes MLSE $\hat{x}_{1:N}$ via forward dynamic programming given HMM obs sequence $y_{1:N}$. For $k = 1, 2, \ldots, N$

$$\delta_{k+1}(j) = \max_i \Big[\delta_k(i) + \log P_{ij}\Big] + \log p(y_{k+1}|x_{k+1} = q_j)$$

$$u_{k+1}(j) = \arg\max_i \Big[\delta_k(i) + \log P_{ij}\Big] + \log p(y_{k+1}|x_{k+1} = q_j)$$

Terminate. $\hat{x}_N = \arg\max_i \delta_N(i)$

Backtrack. $\hat{x}_k = u_{k+1}(\hat{x}_{k+1}),\ k = N-1, \ldots, 1.$

Comparison of Kalman and HMM filter

(i) KF is linear filter: conditional mean is linear in observation. HMM filter is nonlinear filter.

(ii) KF requires Gaussian noise and a linear state space model. In non Gaussian noise, KF is linear min-var estimator. The HMM filter does not require Gaussian noise – it works for any noise density. Also the observation equation does not have to be linear in the Markov state.

(iii) KF is optimal for correlated noise (linearly filtered white noise). HMM filter depends crucially on iid noise v_k

(iv) Both HMM and KF are geometrically ergodic, i.e. they forget their initial condition exponentially fast under reasonable conditions on the model.

(v) **Martingale formulation of HMM**: Let $x_k \in \{e_1, \ldots, e_X\}$ denote states of Markov chain. HMM can be represented as

$$x_{k+1} = P' x_k + w_k$$

$$y_k = C x_k + v_k$$

where w_k is a martingale difference: $\mathbb{E}\{w_k | x_0, x_1, \ldots, x_k\} = 0$.

Since w is not Gaussian, Kalman filter is optimal linear estimator for state x_k of above HMM. HMM filter is optimal (nonlinear) estimator for x_k.

Approximate Filters

For general nonlinear systems – no finite dim filter exists. The following approximations are widely used.

1. Grid approximation: HMM approximation

$$q_{k+1}(x_{k+1}) = p(y_{k+1}|x_{k+1}) \int_{\mathbb{R}^X} p(x_{k+1}|x_k) q_k(x_k) dx_k$$

Discretizing x to the grid $[r_1, \ldots, r_M]$ yields HMM filter

$$q_{k+1}(r_j) = p(y_{k+1}|x_{k+1} = r_j) \sum_{i=1}^{M} p(x_{k+1} = r_j | x_k = r_i) q_k(r_i)$$

$O(M^2)$ computations at each time for M grid points. Approx error: $O(M^{-1/X})$ where X = state dim – suffers from curse of dimensionality.

2. Extended Kalman Filter: Linearize, then run KF. Unscented Kalman filter is more sophisticated.

3. MLSE estimators: Compute state sequence estimate $\arg\max_{x_1,\ldots,x_T} p(Y_T, x_1, \ldots, x_T)$. (e.g. Viterbi algorithm)

4. Basis function Kernel approximations:
(i) Gaussian sum approximations, (ii) Particle filters

Particle filters. Sequential MCMC

Particle filters are a randomized grid sub-optimal algorithm for nonlinear filtering. They use delta function basis approximation

$$p(x_{0:n}|y_{1:n}) \approx \sum_{i=1}^{N} \tilde{\omega}_n^{(i)} \delta(x_{0:n}^{(i)}).$$

The trajectories (positions) $\delta(x_{0:n}^{(i)})$ of the N particles propagate randomly according to state dynamics.
Weights $\tilde{\omega}_n^{(i)}$ are updated via Bayes rule.

Remarks: 1. Cloud of samples approximates a pdf; particle filter uses samples (positions) and scalar weights.

2. *Deterministic Grid.* For $\int_{\mathbb{R}^X}$, given N grid points, approx. error

$$\epsilon = O(N^{-1/X}) \implies N = O(\epsilon^{-X}) = \frac{c}{\epsilon^X}$$

Curse of dimensionality

3. *Particle filter*: Mean square error from CLT is $O(N^{-1})$. Randomization breaks the curse of dimensionality! e.g. if state dim $X = 10$, and N is # grid points:

N	10	100	1000
$N^{-1/X}$	0.794	0.631	0.501
N^{-1}	0.1	0.01	0.001

4. *Bootstrap particle filter:* D.Q Mayne, 1968, Automatica. It was re-invented in 1996. Particle filters are a class of *sequential Markov Chain Monte Carlo (MCMC) algorithms.*

General Particle Filter Algorithm

Consider general model:

$$p(x_{k+1}|x_{0:k}), \quad p(y_k|y_{1:k-1}, x_{1:k})$$

Can have long range dependency in state and observation compared to $p(x_{k+1}|x_k)$ and $p(y_k|x_k)$.

Aim: Estimate $\mathbb{E}\{\phi(x_{0:k})|y_{1:k}\}$ via sequential MCMC.
If we choose $\phi(x_{0:k}) = x_k$, then yields $\mathbb{E}\{x_k|y_{1:k}\}$.

Main idea – Sequential Importance Sampling.
Given samples (trajectories) $x^{(i)}_{0:k-1} \sim p(x_{0:k-1}|y_{1:k-1})$, simulate samples (trajectories) $x^{(i)}_{0:k} \sim p(x_{0:k}|y_{1:k})$ by *re-using* samples $x^{(i)}_{0:k-1}$, $i = 1, \ldots, N$.

1. Simulate $x^{(i)}_k \sim \pi(x_k|x_{0:k-1}, y_{1:k})$.
2. Set $x^{(i)}_{0:k} = (x^{(i)}_{0:k-1}, x^{(i)}_k)$.

How to choose importance sampling density $\pi(x_k|x_{0:k-1}, y_{1:k})$?

General Particle Filter

1. **Generalized Model**: $p(x_{k+1}|x_{0:k}), \quad p(y_k|y_{1:k-1}, x_{1:k})$.
Aim: Simulate trajectories $x_{0:k}^{(i)}$, $i = 1, \ldots, N$ from $p(x_{0:k}|y_{1:k})$ for $k = 1, 2 \ldots$ via sequential MCMC.
Clearly $x_k^{(i)} \sim p(x_k|y_{1:k})$. (no integration needed to marginalize!)

2. **Bayesian Importance sampling**:

$$\mathbb{E}\{\phi(x_{0:k})|y_{1:k}\} = \int \phi(x_{0:k}) \frac{p(x_{0:k}|y_{1:k})}{\pi(x_{0:k}|y_{1:k})} \pi(x_{0:k}|y_{1:k}) dx_{0:k}$$

Sample from importance density: $x_{0:k}^{(i)} \sim \pi(x_{0:k}|y_{1:k})$. From SLLN

$$\sum_{i=1}^{N} \phi(x_{0:k}^{(i)}) \frac{w_k^{(i)}}{\sum_j w_k^{(j)}} \to \mathbb{E}\{\phi(x_{0:k})|y_{1:k}\}, \quad \text{where } w_k^{(i)} = \frac{p(x_{0:k}^{(i)}|y_{1:k})}{\pi(x_{0:k}^{(i)}|y_{1:k})}$$

So density estimate $p(x_{0:k} = x_{0:k}^{(i)}|y_{1:k}) \approx \sum_{i=1}^{N} \delta(x_{0:k}^{(i)}) \frac{w_k^{(i)}}{\sum_{j=1}^{N} w_k^{(j)}}$

3. **Sequential Importance sampling**: (real time)
Aim: Given samples $x_{0:k-1}^{(i)} \sim p(x_{0:k-1}|y_{1:k-1})$, simulate samples $x_{0:k}^{(i)} \sim p(x_{0:k}|y_{1:k})$ by *re-using* samples $x_{0:k-1}^{(i)}$, $i = 1, \ldots, N$.
Note $\pi(x_{0:k}|y_{1:k}) = \pi(x_0|y_{1:k}) \prod_{t=1}^{k} \pi(x_t|x_{0:t-1}, y_{1:k})$
Real time: $\pi(x_{0:k}|y_{1:k}) = \pi(x_0) \prod_{t=1}^{k} \pi(x_t|x_{0:t-1}, y_{1:t})$.

Step (i). Simulate $x_k^{(i)} \sim \pi(x_k|x_{0:k-1}, y_{1:k})$. Set $x_{0:k}^{(i)} = (x_{0:k-1}^{(i)}, x_k^{(i)})$.
Step (ii). Update importance weights in real time: For $i = 1, \ldots, N$,

$$w_k(x_{0:k}^{(i)}) = \frac{p(x_{0:k}^{(i)}|y_{1:k})}{\pi(x_{0:k}^{(i)}|y_{1:k})} \propto \frac{p(y_k|y_{1:k-1}, x_{0:k}^{(i)})\, p(x_k^{(i)}|x_{0:k-1}^{(i)})}{\pi(x_k^{(i)}|x_{0:k-1}^{(i)}, y_{1:k})} \underbrace{\frac{p(x_{0:k-1}^{(i)}|y_{1:k-1})}{\pi(x_{0:k-1}^{(i)}|y_{1:k-1})}}_{w_{k-1}(x_{0:k-1}^{(i)})}$$

Summary: General Particle filter algorithm

Sequential Importance Sampling step: At each time k

- Sample N particles $\tilde{x}_k^{(i)} \sim \pi(x_k|x_{0:k-1}^{(i)}, y_{1:k})$.
  Set $\tilde{x}_{0:k}^{(i)} = (x_{0:k-1}^{(i)}, \tilde{x}_k^{(i)})$.
- Update importance weights w_k and normalized importance weights $\tilde{\omega}_k^{(i)}$ of particles

$$w_k(\tilde{x}_{0:k}^{(i)}) \propto \frac{p(y_k|y_{1:k-1}, \tilde{x}_{0:k}^{(i)}) p(\tilde{x}_k^{(i)}|x_{0:k-1}^{(i)})}{\pi(\tilde{x}_k^{(i)}|x_{0:k-1}^{(i)}, y_{1:k})} w_{k-1}(x_{0:k-1}^{(i)})$$

$$\tilde{\omega}_k^{(i)} = \frac{w_k(\tilde{x}_{0:k}^{(i)})}{\sum_{j=1}^N w_k(\tilde{x}_{0:k}^{(j)})}.$$

Selection step: (optional - to combat degeneracy).
Effective number of particles: $\hat{N} = \frac{1}{\sum_{i=1}^N [\tilde{\omega}_k^{(i)}]^2}$. If $\hat{N}$ is smaller than a prescribed threshold, then

- Multiply/Discard particles $\tilde{x}_{0:k}^{(i)}$, $i = 1, \ldots, N$ with high/low normalised importance weights $\tilde{\omega}_k^{(i)}$ to obtain N new particles $x_{0:k}^{(i)}$, $i = 1, \ldots, N$ with unit weight.

Summary: The particle filter approximation is:

$$p(x_{0:k}|y_{1:k}) \approx \sum_{i=1}^N \tilde{\omega}_k^{(i)} \delta(x_{0:k}^{(i)}), \quad p(x_k|y_{1:k}) \approx \sum_{i=1}^N \tilde{\omega}_k^{(i)} \delta(x_k^{(i)}),$$

$$\mathbb{E}\{x_n|y_{0:n}\} \approx \sum_{i=1}^N \tilde{\omega}_k^{(i)} x_n^{(i)}$$

Example. Bootstrap Particle Filter

Suppose model is: $p(x_{k+1}|x_k), \quad p(y_k|x_k)$
Choose importance density $\pi(x_k|x_{0:k-1}^{(i)}, y_{1:k}) = p(x_k|x_{k-1}^{(i)})$

- Sample N particles $\tilde{x}_k^{(i)} \sim p(x_k|x_{k-1}^{(i)})$.
  Set $\tilde{x}_{0:k}^{(i)} = (x_{0:k-1}^{(i)}, \tilde{x}_k^{(i)})$.
- Importance weight update:

$$w_k(\tilde{x}_{0:k}^{(i)}) \propto \frac{p(y_k|\tilde{x}_k^{(i)}) \cancel{p(\tilde{x}_k^{(i)}|x_{k-1}^{(i)})}}{\cancel{p(\tilde{x}_k^{(i)}|x_{k-1}^{(i)})}} w_{k-1}(x_{k-1}^{(i)})$$

$$\tilde{\omega}_k^{(i)} = \frac{w_k(\tilde{x}_{0:k}^{(i)})}{\sum_{j=1}^N w_k(\tilde{x}_{0:k}^{(j)})}.$$

- Selection step: (optional) Use SIR to resample.
  Sample integers $I^{(i)} \sim \tilde{\omega}_k^{(i)}$
  Then $x_k^{(i)} = \tilde{x}_k^{(I^{(i)})}$, $i = 1, \ldots, N$.
  Each resampled particle has weight $w_k(x_{0:k}^{(i)}) = 1$.

Resampling increases variance but prevents degeneracy.

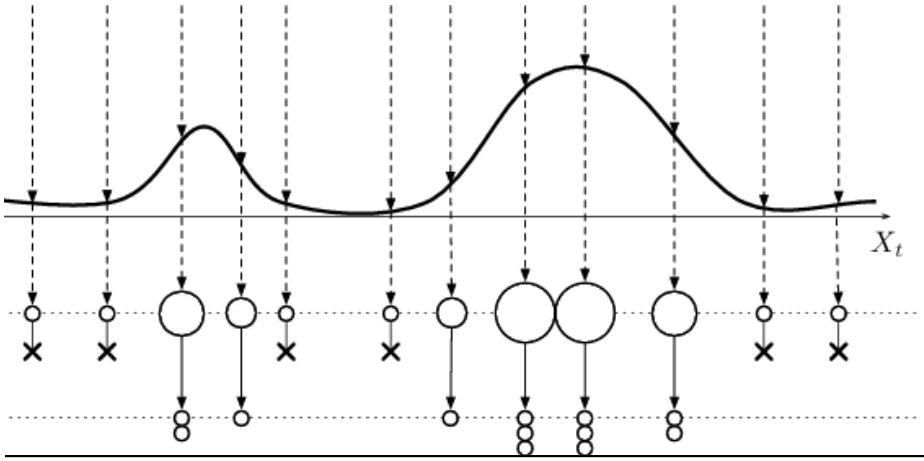
Implementation Issues

1. **Degeneracy**: Variance of importance weights $w_k^{(i)}$ grows with time. Most weights become close to zero – ill-conditioning.

Selection/Resampling Step: (zillions of papers)

(i) Discard particles with low normalized weight.

(ii) Multiply particles with high weight.



Main idea: Sampling Importance Resampling.

Suppose $x^{(i)} \sim \pi(x|y)$ and $w^{(i)} = \frac{p(x^{(i)}|y)}{\pi(x^{(i)}|y)}$, $i = 1, \ldots, N$.

Define normalized weights $\tilde{\omega}^{(i)} = \frac{w^{(i)}}{\sum_{j=1}^N w^{(j)}}$.

Sample integers $I^{(i)} \sim \tilde{\omega}^{(i)}$, $i = 1, \ldots, N$.

Then $x^{(I^{(i)})} \sim p(x|y)$ and each $x^{(I^{(i)})}$ has unit weight.

Before selection: $p(x_{0:k}|y_{0:k}) \propto \sum_i w_k^{(i)} \delta(x_{0:k}^{(i)})$

After selection: $p(x_{0:k}|y_{0:k}) \propto \sum_i \delta(x_{0:k}^{I^{(i)}})$

Selection (resampling) scheme increases variance - so compromise between degeneracy and variance.

2. **Choice of importance function**: Many possibilities:

(i) Bootstrap filter: Choose $\pi(x_k|x_{0:k-1}, y_{0:k}) = p(x_k|x_{k-1})$.
Then $\boxed{w_k^{(i)} = w_{k-1}^{(i)} p(y_k|x_k^{(i)})}$.
If SIR step applied every time, then $w_k^{(i)} = p(y_k|x_k^{(i)})$.
Sensitive to outliers. Particles evolve indpt of obs.

(ii) Fixed importance function: $\pi(x_k|x_{0:k-1}, y_{0:k}) = p(x_k)$

(iii) *Optimal Choice*: Minimizes
$\text{Var}(w_k|y_{1:k}, x_{0:k-1}) = \text{Var}(\frac{p(x_{0:k}|y_{1:k})}{\pi(x_{0:k}|y_{1:k})}|y_{1:k}, x_{0:k-1})$ wrt x_k.

$$\text{Choose} \quad \pi(x_k|x_{0:k-1}, y_{0:k}) = p(x_k|x_{k-1}^{(i)}, y_k)$$

Since $p(x_k|x_{k-1}, y_k) = \frac{p(y_k|x_k)p(x_k|x_{k-1})}{p(y_k|x_{k-1})}$, it follows that

$$\boxed{w_k^{(i)} = w_{k-1}^{(i)}\, p(y_k|x_{k-1}^{(i)})}$$

Compute $w_k^{(i)}$, sample $x_n^{(i)}$ in parallel since indpt of $x_k^{(i)}$!
(a) Need to be able to sample from $p(x_k|x_{k-1}^{(i)}, y_k)$
(b) Need to be able to compute $p(y_k|x_{k-1})$ in closed form.

Example: Nonlinear dynamics, linear observation:

$$x_{k+1} = f(x_k) + v_k, \qquad v_k \sim N(0, \Sigma_v)$$
$$y_k = Cx_k + w_k, \qquad w_k \sim N(0, \Sigma_w)$$

Then $p(x_k|x_{k-1}, y_k) = N(m_k, \Sigma)$ can be computed using a Kalman filter (see my book). Also

$$p(y_k|x_{k-1}) = N\left(Cf(x_{k-1}), (\Sigma_w + C\Sigma_v C')\right)$$