How To Find the Global Maxima (Modes) of a PDF?

- given the function $p(x)$ at left

consider several methods:

1. exhaustive search

   \[
   \text{step} = 1/(\text{iterations}-1); \\
   \text{for } x = 0:\text{step}:1 \\
   \quad \text{if } p(x) > \text{bestp} \\
   \quad \quad \text{bestx} = x; \text{bestp} = p(x); \\
   \quad \text{end} \\
   \text{end}
   \]

   - slow algorithm
   - fast to implement

2. randomized search with uniform sampling

   \[
   \text{while } t < \text{iterations} \\
   x = \text{rand}(1); \\
   \quad \text{if } p(x) > \text{bestp} \\
   \quad \quad \text{bestx} = x; \text{bestp} = p(x); \\
   \quad t = t+1; \% \text{ time} \\
   \text{end}
   \]

   - equally slow algorithm
   - fast to implement

3. random sampling from $p(x)$ (Gibbs sampler)

   - faster algorithm
   - fast to implement but often infeasible (e.g. when $p(x)$ is data dependent (our case in correspondence prob.))

4. Metropolis-Hastings sampling

   - almost as fast (with care)
   - not so fast to implement
   - rarely infeasible
   - RANSAC belongs here
How To Generate Random Samples from a Complex Distribution?

- red: probability density function $\pi(x)$ of the toy distribution on the unit interval

$$\pi(x) = \sum_{i=1}^{4} \gamma_i \text{Be}(x; \alpha_i, \beta_i), \quad \sum_{i=1}^{4} \gamma_i = 1, \quad \gamma_i \geq 0$$

$$\text{Be}(x; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} \cdot x^{\alpha-1}(1-x)^{\beta-1}$$

- alg. for generating samples from $\text{Be}(x; \alpha, \beta)$ is known

- we can generate samples from $\pi(x)$ how?

- suppose we cannot sample from $\pi(x)$ but we can sample from some ‘simple’ distribution $q(x | x_0)$, given the last sample $x_0$ (blue)

$$q(x | x_0) = \begin{cases} 
\text{U}_{0,1}(x) & \text{(independent) uniform sampling} \\
\text{Be}(x; \frac{x_0}{T} + 1, \frac{1-x_0}{T} + 1) & \text{‘beta’ diffusion (crawler)} \\
\pi(x) & \text{ (independent) Gibbs sampler}
\end{cases}$$

- how to redistribute proposal samples $q(x | x_0)$ to target distribution $\pi(x)$ samples?

- note we have unified all the random sampling methods from the previous slide
Metropolis-Hastings (MH) Sampling

$C$ – configuration (of all variable values)  
\[ \text{eg. } C = x \text{ and } \pi(C) = \pi(x) \text{ from } \rightarrow 116 \]

**Goal:** Generate a sequence of random samples \( \{C_t\} \) from target distribution \( \pi(C) \)

- setup a Markov chain with a suitable transition probability to generate the sequence

**Sampling procedure**

1. given \( C_t \), draw a random sample \( S \) from \( q(S \mid C_t) \)

\[ q \text{ may use some information from } C_t \text{ (Hastings)} \]

2. compute acceptance probability

\[ a = \min \left\{ 1, \frac{\pi(S)}{\pi(C_t)} \cdot \frac{q(C_t \mid S)}{q(S \mid C_t)} \right\} \]

3. draw a random number \( u \) from unit-interval uniform distribution \( U_{0,1} \)

4. if \( u \leq a \) then \( C_{t+1} := S \) else \( C_{t+1} := C_t \)

‘Programming’ an MH sampler

1. design a proposal distribution (mixture) \( q \) and a sampler from \( q \)

2. write functions \( q(C_t \mid S) \) and \( q(S \mid C_t) \) that are proper distributions  
\[ \text{not always simple} \]

**Finding the mode**

- remember the best sample  
\[ \text{fast implementation but must wait long to hit the mode} \]

- use simulated annealing  
\[ \text{very slow} \]

- start local optimization from the best sample  
\[ \text{good trade-off between speed and accuracy} \]

an optimal algorithm does not use just the best sample: a Stochastic EM Algorithm (e.g. SAEM)
MH Sampling Demo

- blue point: current sample
- green circle: best sample so far
- histogram: current distribution of visited states
- the vicinity of modes are the most often visited states

sampling process (video, 7:33, 100k samples)

quality = \pi(x)
Demo Source Code (Matlab)

```matlab
function x = proposal_gen(x0)
    % proposal generator q(x | x0)
    T = 0.01; % temperature
    x = betarnd(x0/T+1,(1-x0)/T+1);
end

function p = proposal_q(x, x0)
    % proposal distribution q(x | x0)
    T = 0.01;
    p = betapdf(x, x0/T+1, (1-x0)/T+1);
end

function p = target_p(x)
    % target distribution p(x)
    % shape parameters:
    a = [2 40 100 6];
    b = [10 40 20 1];
    % mixing coefficients:
    w = [1 0.4 0.253 0.50]; w = w/sum(w);
    p = 0;
    for i = 1:length(a)
        p = p + w(i)*betapdf(x,a(i),b(i));
    end
end

%% DEMO script
k = 10000;    % number of samples
X = NaN(1,k); % list of samples
x0 = proposal_gen(0.5);
for i = 1:k
    x1 = proposal_gen(x0);
    a = target_p(x1)/target_p(x0) * ...
        proposal_q(x0,x1)/proposal_q(x1,x0);
    if rand(1) < a
        X(i) = x1; x0 = x1;
    else
        X(i) = x0;
    end
end

depth = 0:0.001:1;
plot(depth, target_p(depth), 'r', 'linewidth',2);
hold on
binw = 0.025; % histogram bin width
n = histc(X, 0:binw:1);
h = bar(0:binw:1, n/sum(n)/binw, 'histc');
set(h, 'facecolor', 'r', 'facealpha', 0.3)
xlim([0 1]); ylim([0 2.5])
xlabel 'x'
ylabel 'p(x)'
title 'MH demo'
hold off
```

3D Computer Vision: V. Optimization for 3D Vision (p. 119/189)
Stripping MH Down

• when we are interested in the best sample only... and we need fast data exploration...

Simplified sampling procedure

1. given $C_t$, draw a random sample $S$ from $q(S \mid C_t) q(S)$ independent sampling
   no use of information from $C_t$

2. compute acceptance probability

   $$a = \min \left\{ 1, \frac{\pi(S)}{\pi(C_t)} \cdot \frac{q(C_t \mid S)}{q(S \mid C_t)} \right\}$$

3. draw a random number $u$ from unit interval uniform distribution $U_{0,1}$

4. if $u \leq a$ then $C_{t+1} := S$ else $C_{t+1} := C_t$

5. if $\pi(S) > \pi(C_{\text{best}})$ then remember $C_{\text{best}} := S$

Steps 2–4 make no difference when waiting for the best sample

• ...but getting a good accuracy sample might take very long this way

• good overall exploration but slow convergence in the vicinity of a mode where $C_t$ could serve as an attractor

• cannot use the past generated samples to estimate any parameters

• we will fix these problems by (possibly robust) 'local optimization'
1. **primitives** = elementary measurements
   - points in line fitting
   - matches in epipolar geometry estimation

2. configuration = *s*-tuple of primitives
   minimal subsets necessary for parameter estimate
   - the minimization will be over a discrete set:
     - of point pairs in line fitting (left)
     - of match 7-tuples in epipolar geometry estimation

3. proposal distribution \( q(\cdot) \) is then given by the empirical distribution of *s*-tuples:
   a) propose *s*-tuple from data independently \( q(S \mid C^t) = q(S) \)
      i) \( q \) uniform \( q(S) = \left( \frac{mn}{s} \right)^{-1} \)
      ii) \( q \) dependent on descriptor similarity
   b) solve the minimal geometric problem \( \mapsto \) parameter proposal
   - pairs of points define line distribution from \( p(\mathbf{n} \mid X) \) (left)
   - random correspondence tuples drawn uniformly propose samples of \( \mathbf{F} \) from a data-driven distribution \( q(\mathbf{F} \mid M) \)

4. local optimization from promising proposals

5. stopping based on the probability of mode-hitting
RANSAC with Local Optimization and Early Stopping

1. initialize the best sample as empty $C_{\text{best}} := \emptyset$ and time $t := 0$
2. estimate the number of needed proposals as $N := \binom{n}{s}$
   $n$ – No. of primitives, $s$ – minimal sample size
3. while $t \leq N$:
   a) propose a minimal random sample $S$ of size $s$ from $q(S)$
   b) if $\pi(S) > \pi(C_{\text{best}})$ then
      i) update the best sample $C_{\text{best}} := S$
      ii) threshold-out inliers using $e_T$ from (27)
   iii) start local optimization from the inliers of $C_{\text{best}}$
        LM optimization with robustified (→113) Sampson error
        possibly weighted by posterior $\pi(m_{ij})$ [Chum et al. 2003]
   iv) update $C_{\text{best}}$, update inliers using (27), re-estimate $N$ from inlier counts
   c) $t := t + 1$
4. output $C_{\text{best}}$

• see MPV course for RANSAC details
see also [Fischler & Bolles 1981], [25 years of RANSAC]
**Principle:** what is the number of proposals \( N \) that are needed to hit an all-inlier sample?

\[
N \geq \frac{\log(1 - P)}{\log(1 - \varepsilon^s)}
\]

- \( P \) ... probability that at least one proposal is an all-inlier
- \( \varepsilon \) ... the fraction of inliers among primitives, \( \varepsilon \leq 1 \)
- \( s \) ... minimal sample size (2 in line fitting, 7 in 7-point algorithm)

- \( \varepsilon^s \) ... proposal does not contain an outlier
- \( 1 - \varepsilon^s \) ... proposal contains at least one outlier
- \( (1 - \varepsilon^s)^N \) ... \( N \) previous proposals contained an outlier = \( 1 - P \)

\[
1 - P = (1 - \varepsilon^s)^N
\]

\( N \) for \( s = 7 \)

<table>
<thead>
<tr>
<th>( \varepsilon )</th>
<th>( P = 0.5 )</th>
<th>( P = 0.8 )</th>
<th>( P = 0.99 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>205</td>
<td>590</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>( 1.3 \cdot 10^5 )</td>
<td>( 3.5 \cdot 10^5 )</td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>( 1.6 \cdot 10^7 )</td>
<td>( 4.6 \cdot 10^7 )</td>
<td></td>
</tr>
</tbody>
</table>

\( N \) can be re-estimated using the current estimate for \( \varepsilon \) (if there is LO, then after LO)

the quasi-posterior estimate for \( \varepsilon \) is the average over all samples generated so far

- this shows we have a good reason to limit all possible matches to tentative matches only
- for \( \varepsilon \to 0 \) we gain nothing over the standard MH-sampler stopping criterion
Example Matching Results for the 7-point Algorithm with RANSAC

- notice some wrong matches (they have wrong depth, even negative)
- they cannot be rejected without additional constraints or scene knowledge
- without local optimization the minimization is over a discrete set of epipolar geometries proposable from 7-tuples
Thank You