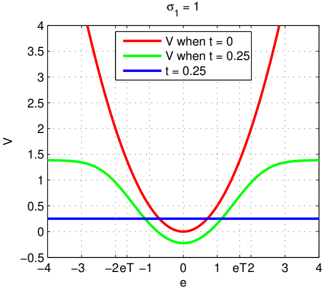
# ▶The Action of the Robust Matching Model on Data

## Example for $\hat{V}(e_{ij})$ from (26):




red – the (non-robust) quadratic error $\hat{V}(e_{ij})$ when $t = 0$

blue – the rejected match penalty $t$

green – robust $\hat{V}(e_{ij})$ from (26)

- if the error of a correspondence exceeds a limit, it is ignored
- then $\hat{V}(e_{ij}) = \text{const}$ and we just count outliers in (26)
- $t$ controls the 'turn-off' point
- the inlier/outlier threshold is $e_T$ – the error for which $(1 - P_0)\, p_1(e_T) = P_0\, p_0(e_T)$: note that $t \approx 0$

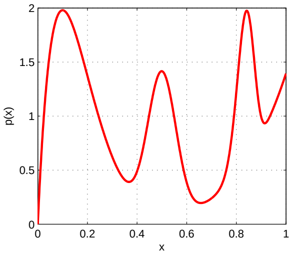
$$e_T = \sigma_1 \sqrt{-\log t^2}, \quad t = e^{-\frac{1}{2}\left(\frac{e_T}{\sigma_1}\right)^2} \tag{27}$$

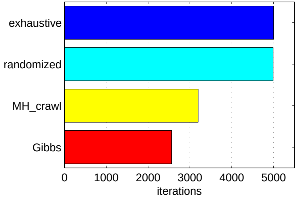
## The full optimization problem (23) uses (26):

$$\mathbf{F}^* = \arg\max_{\mathbf{F}} \frac{\overbrace{p(E, D \mid \mathbf{F})}^{\text{data model}} \cdot \overbrace{p(\mathbf{F})}^{\text{prior}}}{\underbrace{p(E, D)}_{\text{evidence}}} \approx \arg\min_{\mathbf{F}} \left[ V(\mathbf{F}) + \sum_{i=1}^{m} \sum_{j=1}^{n} \log\left( e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}} + t \right) \right]$$

- $\pi(\mathbf{F})$ – a shorthand for the argument of the maximization
- typically we take $V(\mathbf{F}) = -\log p(\mathbf{F}) = 0$ unless we need to stabilize a computation, e.g. when video camera moves smoothly (on a high-mass vehicle) and we have a prediction for $\mathbf{F}$
- evidence is not needed unless we want to compare different models (e.g. homography vs. epipolar geometry)

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




- averaged over $10^4$ trials
- number of proposals before $|x - x_{\text{true}}| \le \text{step}$

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

p.d.f. on $[0, 1]$, mode at $0.1$

**consider several methods:**

1. exhaustive search

```
step = 1/(iterations-1);
for x = 0:step:1
 if p(x) > bestp
  bestx = x; bestp = p(x);
 end
end
```

- slow algorithm (definite quantization)
- fast to implement

2. randomized search with uniform sampling

```
while t < iterations
 x = rand(1);
 if p(x) > bestp
  bestx = x; bestp = p(x);
 end
 t = t+1;  % time
end
```

- equally slow algorithm
- fast to implement

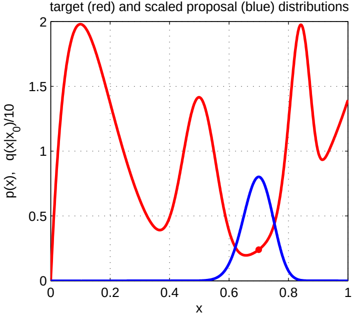
3. random sampling from $p(x)$ (Gibbs sampler)

• 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 target distribution

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

$p(x|g)\,p(g)$

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

- alg. for generating samples from $\mathrm{Be}(x; \alpha, \beta)$ is known
- $\Rightarrow$ we can generate samples from $\pi(x)$ how?

- suppose we cannot sample from $\pi(x)$ but we can sample from some 'simple' proposal distribution $q(x \mid x_0)$, given the previous sample $x_0$ (blue)

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

- note we have unified all the random sampling methods from the previous slide
- how to redistribute proposal samples $q(x \mid x_0)$ to target distribution $\pi(x)$ samples?

# ▶Metropolis-Hastings (MH) Sampling

$C$ – configuration (of all variable values) e.g. $C = x$ and $\pi(C) = \pi(x)$ from →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 $q(S \mid C_t)$

   $q$ may use some information from $C_t$ (Hastings)

2. compute acceptance probability

   the evidence term drops out

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

3. draw a random number $u$ from unit-interval uniform distribution $\mathrm{U}_{0,1}$
4. if $u \le a$ 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

   not always simple

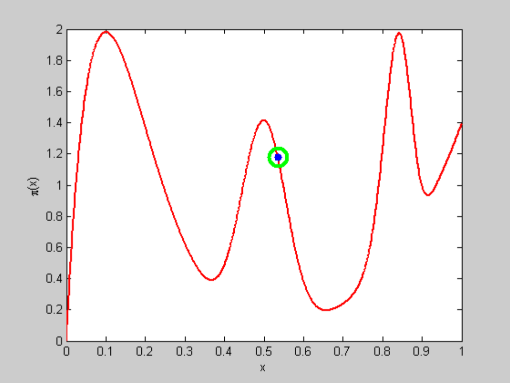
## Finding the mode

$\pi(x)^{\frac{1}{T}}$

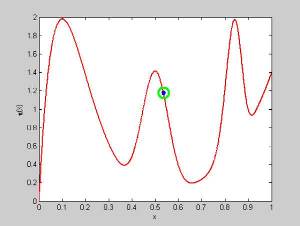
- remember the best sample — fast implementation but must wait long to hit the mode
- use simulated annealing — very slow
- start local optimization from the best sample — good trade-off between speed and accuracy

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

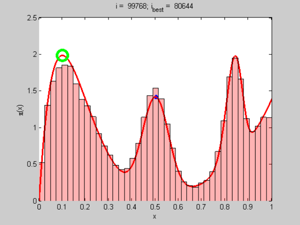
# MH Sampling Demo




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




initial sample




final distribution of visited states

- blue point: current sample
- green circle: best sample so far     quality $= \pi(x)$
- histogram: current distribution of visited states
- the vicinity of modes are the most often visited states

# Demo Source Code (Matlab)

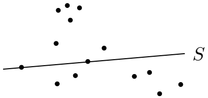
```
function x = proposal_gen(x0)
% proposal generator q(x | x0)

 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

figure(1)
x = 0:0.001:1;
plot(x, target_p(x), '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
```

# ▶The Elements of a Data-Driven MH Sampler

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. a map from configuration $C$ to parameters $\boldsymbol{\theta}$ by solving the minimal geometric problem
   - line parameters $\mathbf{n}$ from two points
   - fundamental matrix $\mathbf{F}$ from seven matches

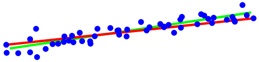
4. target likelihood $p(E, D \mid \boldsymbol{\theta})$ replaces $\pi(C)$
   - can use log-likelihood: then it is the sum of robust errors $\hat{V}(e_{ij})$ given $\mathbf{F}$ (26)
   - robustified point distance from the line $\boldsymbol{\theta} = \mathbf{n}$
   - robustified Sampson error for $\boldsymbol{\theta} = \mathbf{F}$
   - posterior likelihood $p(E, D \mid \boldsymbol{\theta})p(\boldsymbol{\theta})$ can be used    MAPSAC ($\pi(S)$ includes the prior)

# ▶cont'd

5. (optional) hard inlier/outlier discrimination by the threshold (27)

$$\hat{V}(e_{ij}) < e_T, \qquad e_T = \sigma_1 \sqrt{-\log t^2}$$

6. parameter distribution follows the empirical distribution of $s$-tuples. Since the proposal is done via the minimal problem solver, it is 'data-driven',



- pairs of points define line distribution $p(\mathbf{n} \mid X)$ (left)
- random correspondence 7-tuples define epipolar geometry distribution $q(\mathbf{F} \mid M)$

7. proposal distribution $q(\cdot)$ is just a distribution of the $s$-tuples:
   a) $q$ uniform, independent $q(S \mid C_t) = q(S) = \binom{mn}{s}^{-1}$, then $a = \min\left\{1, \frac{p(S)}{p(C_t)}\right\}$
   b) $q$ dependent on descriptor similarity — PROSAC (similar pairs are proposed more often)
   c) $q$ dependent on the current configuration — e.g. 'not far from it'

8. local optimization from promising proposals
   - can use hard inliers
   - cannot be used to replace $C_t$

9. stopping based on the probability of proposing an all-inlier sample →122

# ▶Data-Driven Sampler Stopping

**Principle:** what is the number of proposals $N$ that are needed to hit an all-inlier sample?

this will tell us nothing about the accuracy of the result

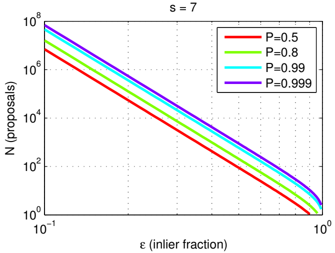
$P$ ...probability that at least one proposal is an all-inlier $1-P$ ...all previous $N$ proposals were bad
$\varepsilon$ ...the fraction of inliers among primitives, $\varepsilon \leq 1$
$s$ ...minimal sample size (2 in line fitting, 7 in 7-point algorithm)

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

- $\varepsilon^s$ ...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 $= 1-P$

$N$ for $s=7$

| | $P$ | |
|---|---|---|
| $\varepsilon$ | 0.8 | 0.99 |
| 0.5 | 205 | 590 |
| 0.2 | $1.3\cdot10^5$ | $3.5\cdot10^5$ |
| 0.1 | $1.6\cdot10^7$ | $4.6\cdot10^7$ |




- $N$ 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

# ▶Stripping MH Down To Get RANSAC [Fischler & Bolles 1981]

- 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 $\cancel{q(S \mid C_t)}$ $q(S)$ — 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 $\mathrm{U}_{0,1}$~~
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 $C_{\text{best}} := S$

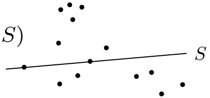
Steps 2–4 make no difference when waiting for the best sample

- ...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'

# ►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 $t \le N$:

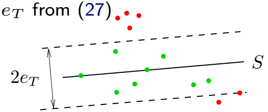
   a) propose a minimal random sample $S$ of size $s$ from $q(S)$



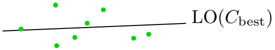

   b) if $\pi(S) > \pi(C_{\text{best}})$ then

      i) update the best sample $C_{\text{best}} := S$ $\pi(S)$ marginalized as in (26); $\pi(S)$ includes a prior $\Rightarrow$ MAP

      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 →122 for derivation

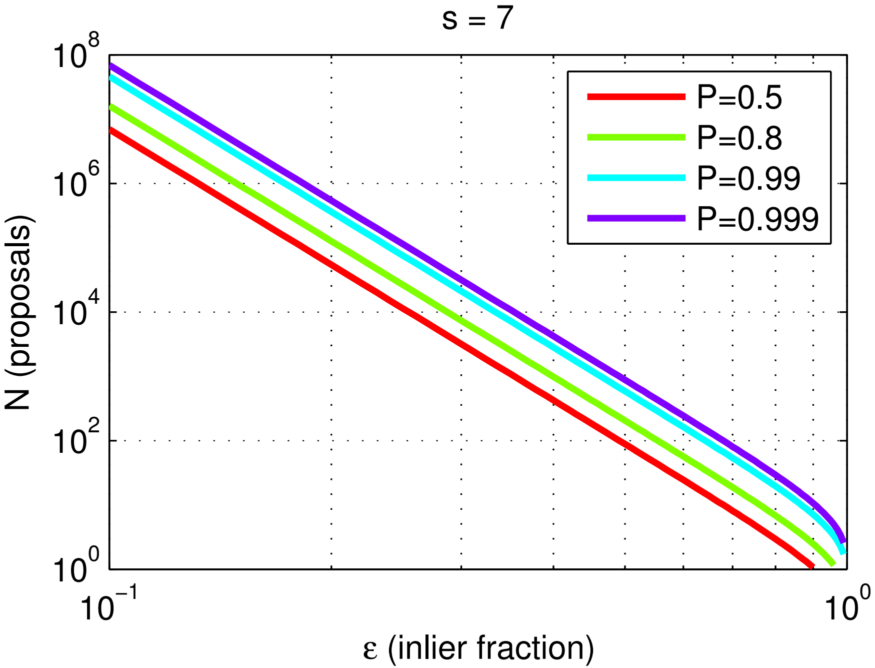
$$N = \frac{\log(1-P)}{\log(1-\varepsilon^s)}, \quad \varepsilon = \frac{|\,\text{inliers}(C_{\text{best}})|}{m\,n},$$

   c) $t := t + 1$

4. output $C_{\text{best}}$

- see MPV course for RANSAC details see also [Fischler & Bolles 1981], [25 years of RANSAC]

Thank You


s = 7
N (proposals)
ε (inlier fraction)
P=0.5
P=0.8
P=0.99
P=0.999
$10^8$
$10^6$
$10^4$
$10^2$
$10^0$
$10^{-1}$
$10^0$