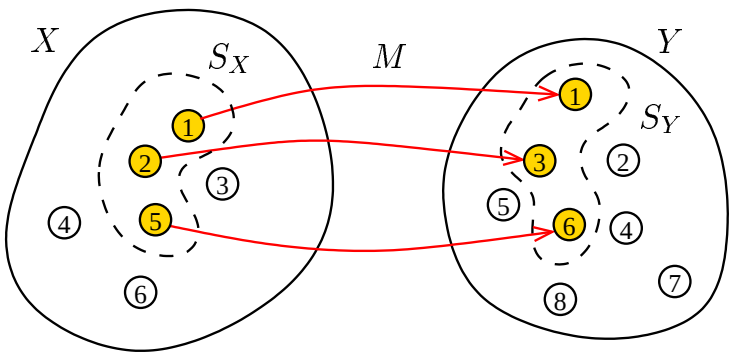
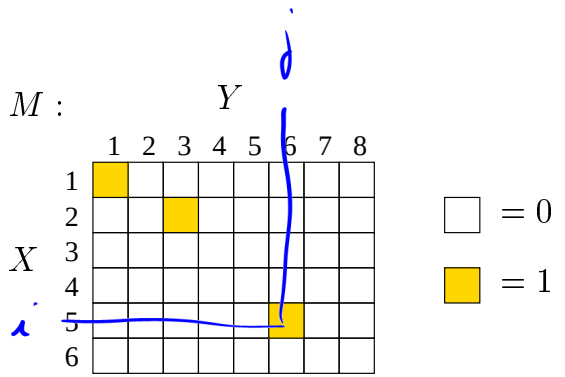
OPPA European Social Fund
Prague & EU: We invest in your future.

▶The Full Problem of Matching and Fundamental Matrix Estimation

Problem: Given two sets of image points $X = \{x_i\}_{i=1}^m$ and $Y = \{y_j\}_{j=1}^n$ and their descriptors D, find the most probable

1. inliers $S_X \subseteq X$, $S_Y \subseteq Y$
2. one-to-one perfect matching $M : S_X \to S_Y$ — perfect matching: 1-factor of the bipartite graph
3. fundamental matrix $\mathbf{F}$ such that $\operatorname{rank} \mathbf{F} = 2$
4. such that for each $x_i \in S_X$ and $y_j = M(x_i)$ it is probable that
   a. the image descriptor $D(x_i)$ is similar to $D(y_j)$, and
   b. the total geometric error $\sum_{ij} e_{ij}^2(\mathbf{F})$ is small — note a slight change in notation: e_{ij}
5. inlier-outlier and outlier-outlier matches are improbable





$$(M^*, \mathbf{F}^*) = \arg\max_{M,\mathbf{F}} p(M, \mathbf{F} \mid X, Y, D) \tag{17}$$

- probabilistic model: an efficient language for task formulation
- the (17) is a p.d.f. for all the involved variables (there is a constant number of variables!)
- binary matching table $M_{ij} \in \{0, 1\}$ of fixed size $m \times n$
  - each row/column contains at most one unity
  - zero rows/columns correspond to unmatched point x_i / y_j

Deriving A Robust Matching Model by Marginalization

For algorithmic efficiency, instead of $(M^*, \mathbf{F}^*) = \arg\max_{M,\mathbf{F}} p(M, \mathbf{F} \mid X, Y, D)$ we will solve

$$\mathbf{F}^* = \arg\max_{\mathbf{F}} p(\mathbf{F} \mid X, Y, D) \tag{18}$$

by marginalization of $p(M, \mathbf{F} \mid X, Y, D)$ over M

this simplification changes the problem!

$$p(M, \mathbf{F} \mid X, Y, D) \simeq p(M, \mathbf{F}, X, Y, D) = p(X, Y, D, M \mid \mathbf{F}) \cdot p(\mathbf{F})$$

assuming correspondence-wise independence:

$$p(X, Y, D, M \mid \mathbf{F}) = \prod_{i=1}^{m} \prod_{j=1}^{n} p(x_i, y_j, D, m_{ij} \mid \mathbf{F}) \overset{\text{def}}{=} \prod_{i=1}^{m} \prod_{j=1}^{n} p_e(e_{ij}, d_{ij}, m_{ij} \mid \mathbf{F})$$

- e_{ij} represents geometric error for match $x_i \leftrightarrow y_i$: $e_{ij}(x_i, y_i \mid \mathbf{F})$
- d_{ij} represents descriptor similarity for match $x_i \leftrightarrow y_i$: $d_{ij} = \|\mathbf{d}(x_i) - \mathbf{d}(y_j)\|$ (SIFT etc)

Marginalization:

$$\sum_{m_{11} \in \{0,1\}} \sum_{m_{12}} \cdots \sum_{m_{mn}} p(X, Y, D, M \mid \mathbf{F}) = \sum_{m_{11}} \sum_{m_{12}} \cdots \sum_{m_{mn}} \prod_{i=1}^{m} \prod_{j=1}^{n} p_e(e_{ij}, d_{ij}, m_{ij} \mid \mathbf{F}) =$$

$$= \cdots = \prod_{i=1}^{m} \prod_{j=1}^{n} \underbrace{\sum_{m_{ij} \in \{0,1\}} p_e(e_{ij}, d_{ij}, m_{ij} \mid \mathbf{F})}_{\text{we will continue with this term}} = p(X, Y, D \mid \mathbf{F})$$

Robust Matching Model (cont'd)

$$\sum_{m_{ij}\in\{0,1\}} p_e(e_{ij}, d_{ij}, m_{ij} \mid \mathbf{F}) = \sum_{m_{ij}\in\{0,1\}} p_e(e_{ij}, d_{ij} \mid m_{ij}, \mathbf{F}) \cdot p(m_{ij} \mid \mathbf{F}) =$$

$$= \underbrace{p_e(e_{ij}, d_{ij} \mid m_{ij}=1, \mathbf{F})}_{p_1(e_{ij},d_{ij}|\mathbf{F})} \cdot \underbrace{p(m_{ij}=1 \mid \mathbf{F})}_{1-\alpha_0} + \underbrace{p_e(e_{ij}, d_{ij} \mid m_{ij}=0, \mathbf{F})}_{p_0(e_{ij},d_{ij}|\mathbf{F})} \cdot \underbrace{p(m_{ij}=0 \mid \mathbf{F})}_{\alpha_0} =$$

$$= (1-\alpha_0)\left[p_1(e_{ij}, d_{ij} \mid \mathbf{F}) + \frac{\alpha_0}{1-\alpha_0} p_0(e_{ij}, d_{ij} \mid \mathbf{F})\right] \quad (19)$$

- the $p_0(e_{ij}, d_{ij} \mid \mathbf{F}) \approx \text{const}$ is a penalty for 'missing a correspondence' but it should be a p.d.f. (cannot be a constant) (see Slide 108 for a simplification)

$$\alpha_0 \to 1, \quad p_0 \to 0 \quad \text{so that} \quad \frac{\alpha_0}{1-\alpha_0} p_0 \approx \text{const}$$

- the $p_1(e_{ij}, d_{ij} \mid \mathbf{F})$ is typically an easy-to-design component: assuming independence of geometric error and descriptor similarity:

$$p_1(e_{ij}, d_{ij} \mid \mathbf{F}) = p_1(e_{ij} \mid \mathbf{F}) \cdot p_1(d_{ij})$$

- we choose, eg.

$$p_1(e_{ij} \mid \mathbf{F}) = \frac{1}{T_e(\sigma_1, \mathbf{F})} e^{-\frac{e_{ij}^2(\mathbf{F})}{2{\sigma_1}^2}}, \quad p_1(d_{ij}) = \frac{1}{T_d(\sigma_d, \dim \mathbf{d})} e^{-\frac{\|\mathbf{d}(x_i)-\mathbf{d}(y_j)\|^2}{2{\sigma_d}^2}} \quad (20)$$

- σ_1, σ_d, α_0 are 'hyper-parameters'
- the form of $T(\sigma_1, \mathbf{F})$ depends on error definition
- we will continue with the result from (19)

▶Simplified Robust Energy (Error) Function

- assuming the choice of p_1 as in (20), we are simplifying

$$p(X, Y, D \mid \mathbf{F}) = \prod_{i=1}^{m} \prod_{j=1}^{n} \Big[(1 - \alpha_0)\, p_1(e_{ij}, d_{ij} \mid \mathbf{F}) + \alpha_0\, p_0(e_{ij}, d_{ij} \mid \mathbf{F}) \Big] \tag{21}$$

- we define 'energy' as: $V(x) = -\log p(x)$ this helps simplify the formulas
- for simplicity, we omit d_{ij}
- we choose $\sigma_0 \gg \sigma_1$ and the missed-correspondence penalty function as

$$p_0(e_{ij} \mid \mathbf{F}) = \frac{1}{T_e(\sigma_0, \mathbf{F})}\, e^{-\frac{e_{ij}^2(\mathbf{F})}{2{\sigma_0}^2}}$$

- then

$$V(X, Y, D \mid \mathbf{F}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \Bigg[\underbrace{-\log \frac{1 - \alpha_0}{T_e(\sigma_1, \mathbf{F})}}_{\Delta(\mathbf{F})} - \log\Big(e^{-\frac{e_{ij}^2(\mathbf{F})}{2{\sigma_1}^2}} + \underbrace{\frac{\alpha_0}{1 - \alpha_0} \frac{T_e(\sigma_1, \mathbf{F})}{T_e(\sigma_0, \mathbf{F})}\, e^{-\frac{e_{ij}^2(\mathbf{F})}{2{\sigma_0}^2}}}_{t \approx \text{const}} \Big) \Bigg]$$

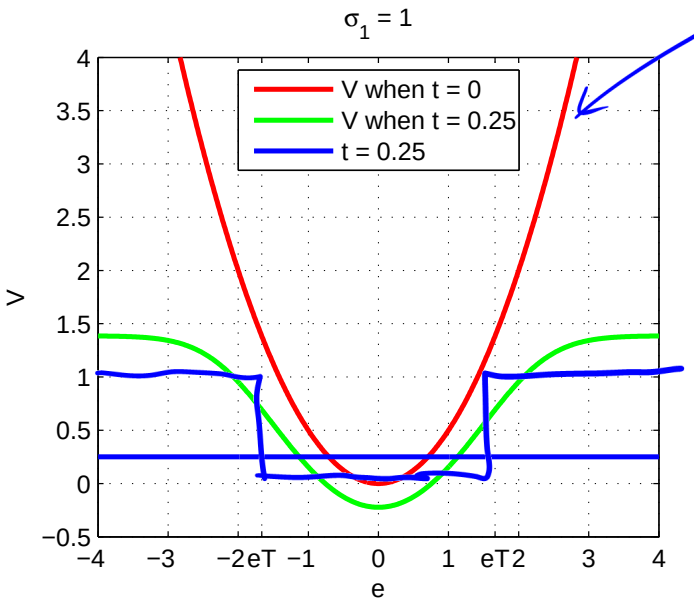
- by choosing representative of $\mathbf{F}$ such that $\Delta(\mathbf{F}) = \text{const}$, we get

$$V(X, Y, D \mid \mathbf{F}) = m\, n\, \Delta + \sum_{i=1}^{m} \sum_{j=1}^{n} \underbrace{-\log\Big(e^{-\frac{e_{ij}^2(\mathbf{F})}{2{\sigma_1}^2}} + t \Big)}_{\hat{V}(e_{ij})} \tag{22}$$

note that m, n are fixed

▶The Action of the Robust Matching Model on Data

Example for $\hat{V}(e)$ from (22):



red – the usual (non-robust) error when $t = 0$

blue – the rejected correspondence penalty t

green – 'robust energy' (22)

- if the error of a correspondence exceeds a limit, it is ignored
- then $\hat{V}(e) = \text{const}$ and we essentially count outliers in (22)
- t controls the 'turn-off' point
- the inlier/outlier threshold is e_T is the error for which $(1 - \alpha_0)\, p_1(e_T) = \alpha_0\, p_0(e_T)$: note that $t \approx 0$

$$e_T = \sigma_1 \sqrt{-\log t^2} \qquad (23)$$

The full optimization problem is (18):

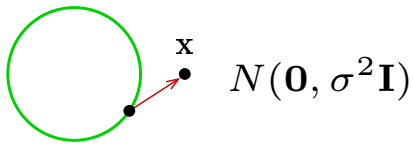
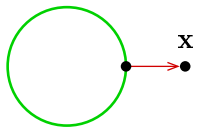
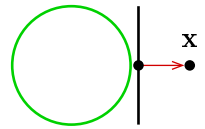
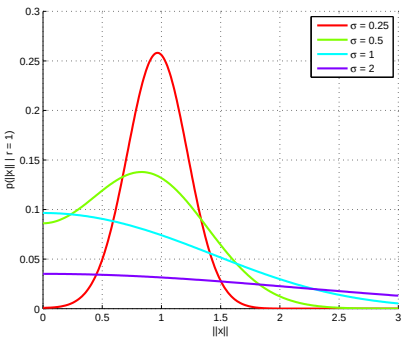
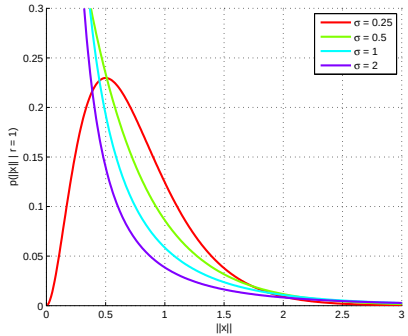
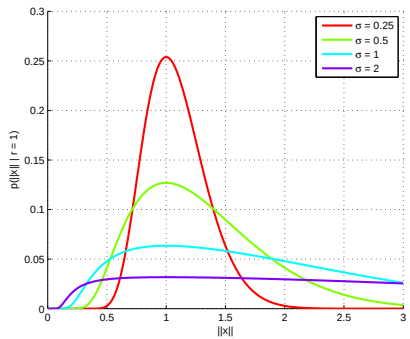
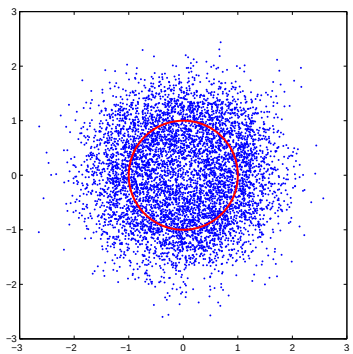
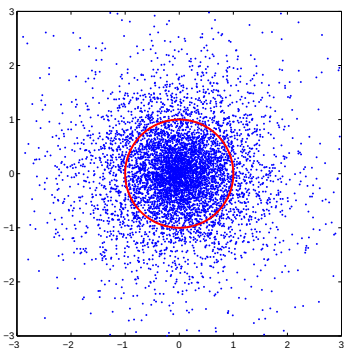
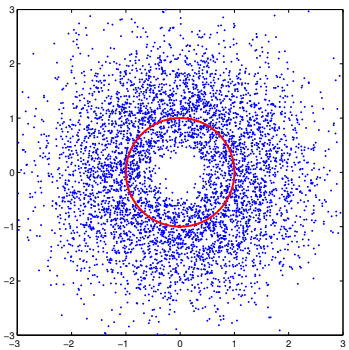
$$\mathbf{F}^* = \arg\max_{\mathbf{F}} p(\mathbf{F} \mid X, Y, D) = \arg\max_{\mathbf{F}} \frac{\overbrace{p(X, Y, D \mid \mathbf{F})}^{\text{likelihood}} \cdot \overbrace{p(\mathbf{F})}^{\text{prior}}}{\underbrace{p(X, Y, D)}_{\text{evidence}}} =$$

$$= \arg\min_{\mathbf{F}} \{ V(X, Y, D \mid \mathbf{F}) + V(\mathbf{F}) \}$$

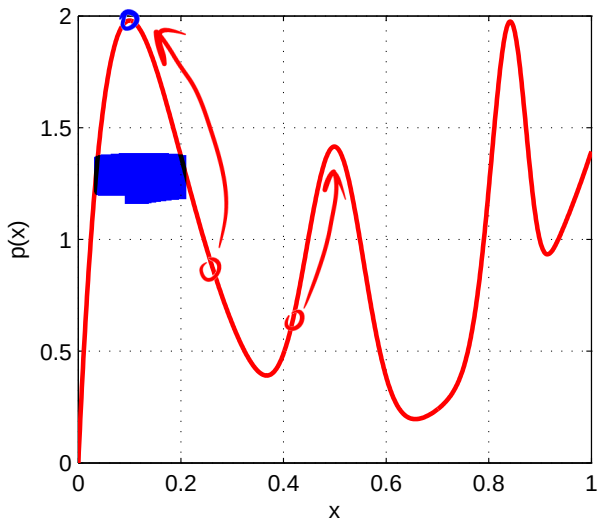
- typically we take $V(\mathbf{F}) = 0$ unless we need to stabilize a computation, e.g. when video camera moves smoothly (on a high-mass vehicle) and we have a prediction for $\mathbf{F}$
- evidence is not needed unless we want to compare different models

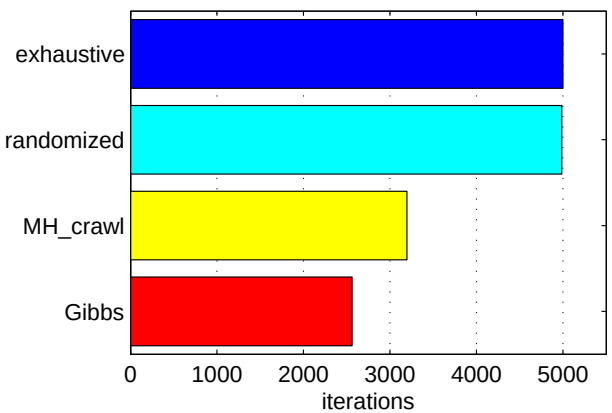
Discussion: On The Art of Probabilistic Model Design...

- a few models for fitting zero-centered circle C of radius r to points in $\mathbb{R}^2$

	marginalized over C	orthogonal deviation from C	Sampson approximation
error model	 $N(\mathbf{0}, \sigma^2\mathbf{I})$	 $\Gamma(\cdot,\cdot)$	 $N(\mathbf{0}, \sigma^2\mathbf{I})$
radial p.d.f.			
random sample			
$p(\mathbf{x} \mid r)$	$\approx \frac{1}{\sigma\sqrt{(2\pi)^3 r\,\|\mathbf{x}\|}}\, e^{-\frac{(\|\mathbf{x}\|-r)^2}{2\sigma^2}}$	$\frac{1}{2\pi\Gamma(\frac{r^2}{\sigma})}\,\frac{1}{\|\mathbf{x}\|^2}\left(\frac{r\|\mathbf{x}\|}{\sigma}\right)^{\frac{r^2}{\sigma}} e^{-\frac{r\|\mathbf{x}\|}{\sigma}}$	$\frac{1}{r\sigma\sqrt{(2\pi)^3}}\, e^{-\frac{e^2(\mathbf{x};r)}{2\sigma^2}}$
	• mode inside the circle • models the inside well • tends to normal distrib.	• peak at the center • unusable for small radii • tends to Dirac distrib.	• mode at the circle • hole at the center • tends to normal distrib.

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





- averaged over 10^4 trials
- number of proposals before $|x - x_{\text{true}}| \leq \text{step}$
- uniform and Gibbs give the theoretical result

- consider the function $p(x)$ at left p.d.f. on $[0, 1]$, mode at 0.1
- consider several methods:

1. exhaustive search

$\arg\max_x p(x)$

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

• slow algorithm (definite quantization); faster variants exist • fast to implement

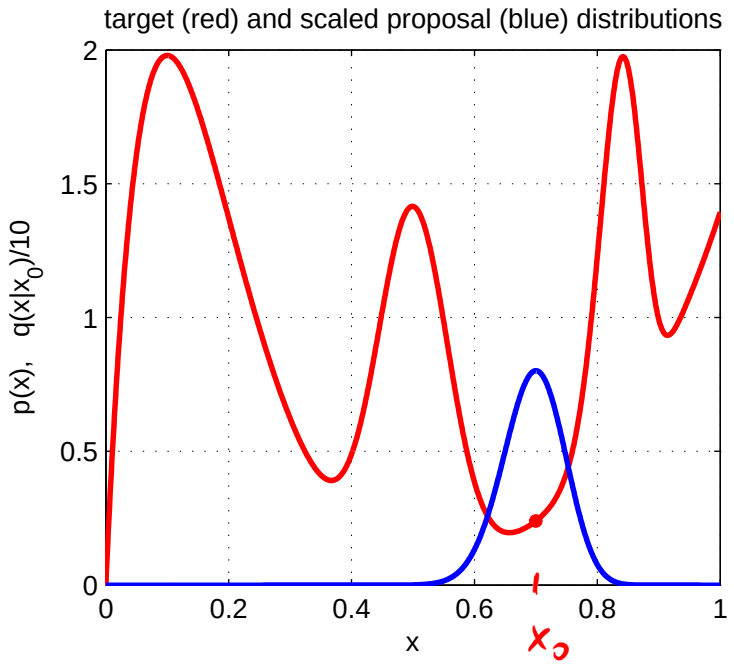
2. randomized search with uniform sampling

```
x = rand(1); x= prnd;
if p(x) > bestp
 bestx = x; bestp = p(x);
end
```

• slow algorithm but better convergence • fast to implement • how to stop it?

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))
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 $p(x)$ of a toy distribution on the unit interval target distribution

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

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

$(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ $(0.1, 0.2, 0.5, 0.2)$ | $(0.1, 0.3, 0.8, 1.0)$

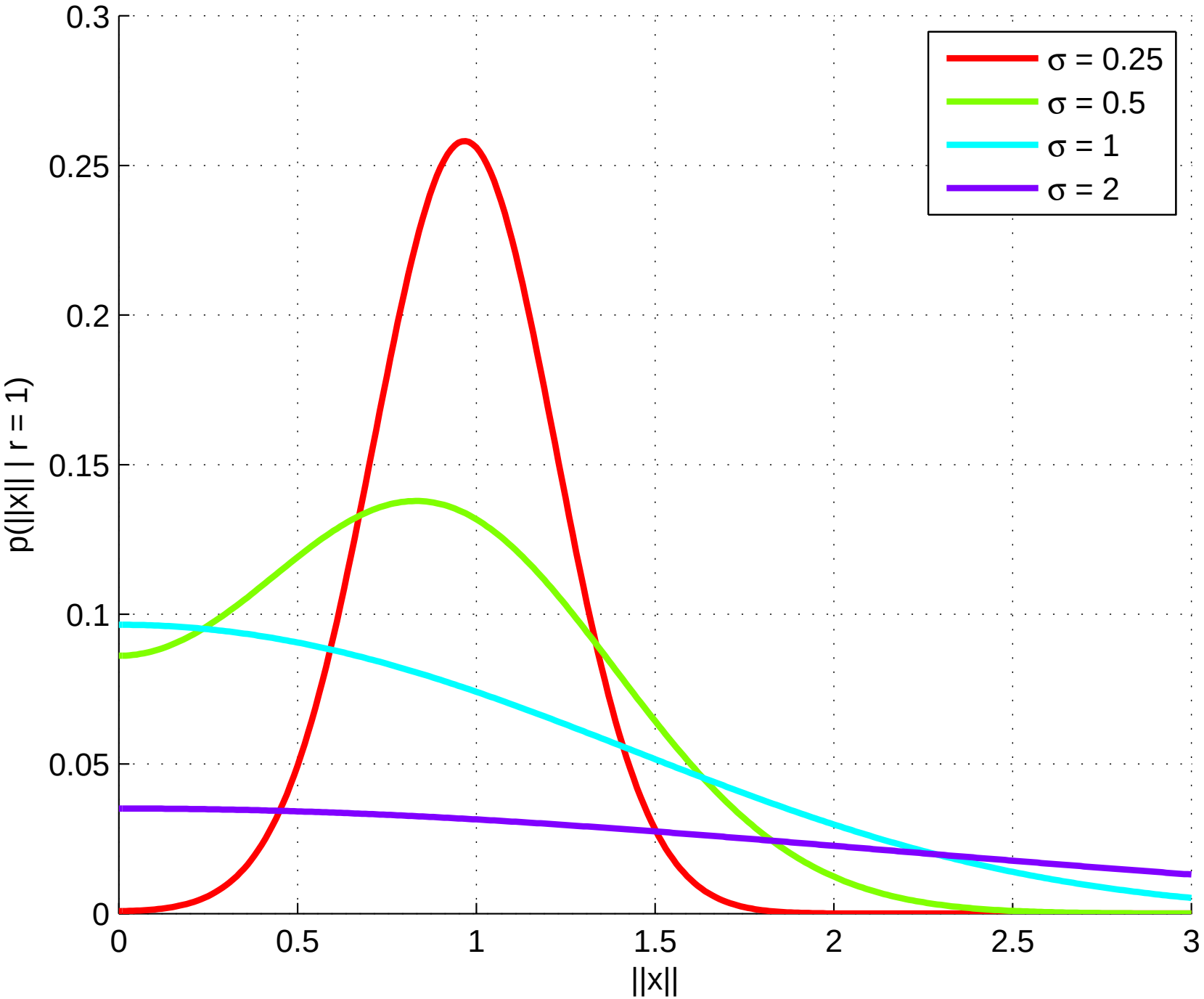
- note we can generate samples from this $p(x)$ how?

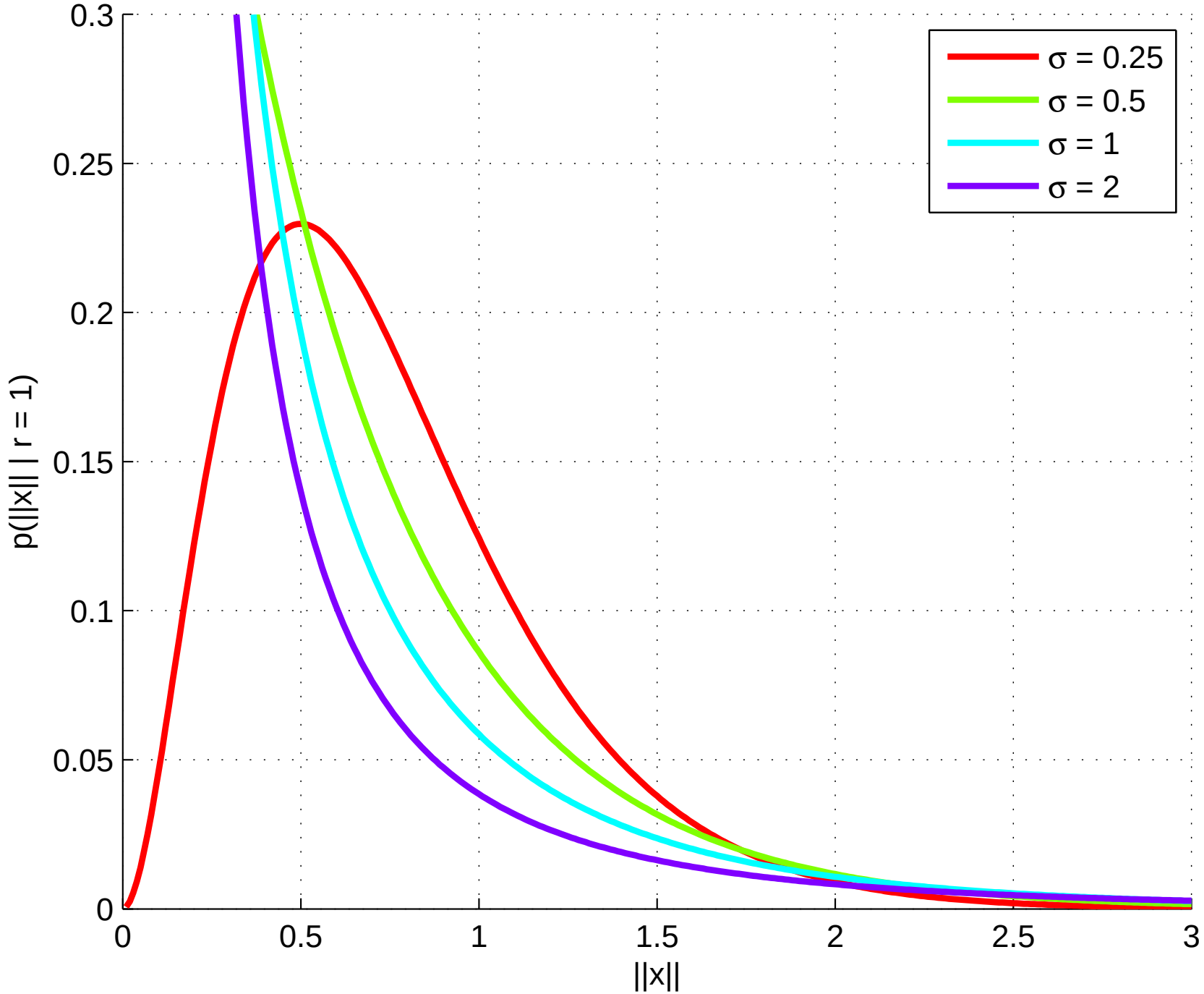
- suppose we cannot sample from $p(x)$ but we can sample from some 'simple' distribution, given the last sample x_0 (blue) proposal distribution

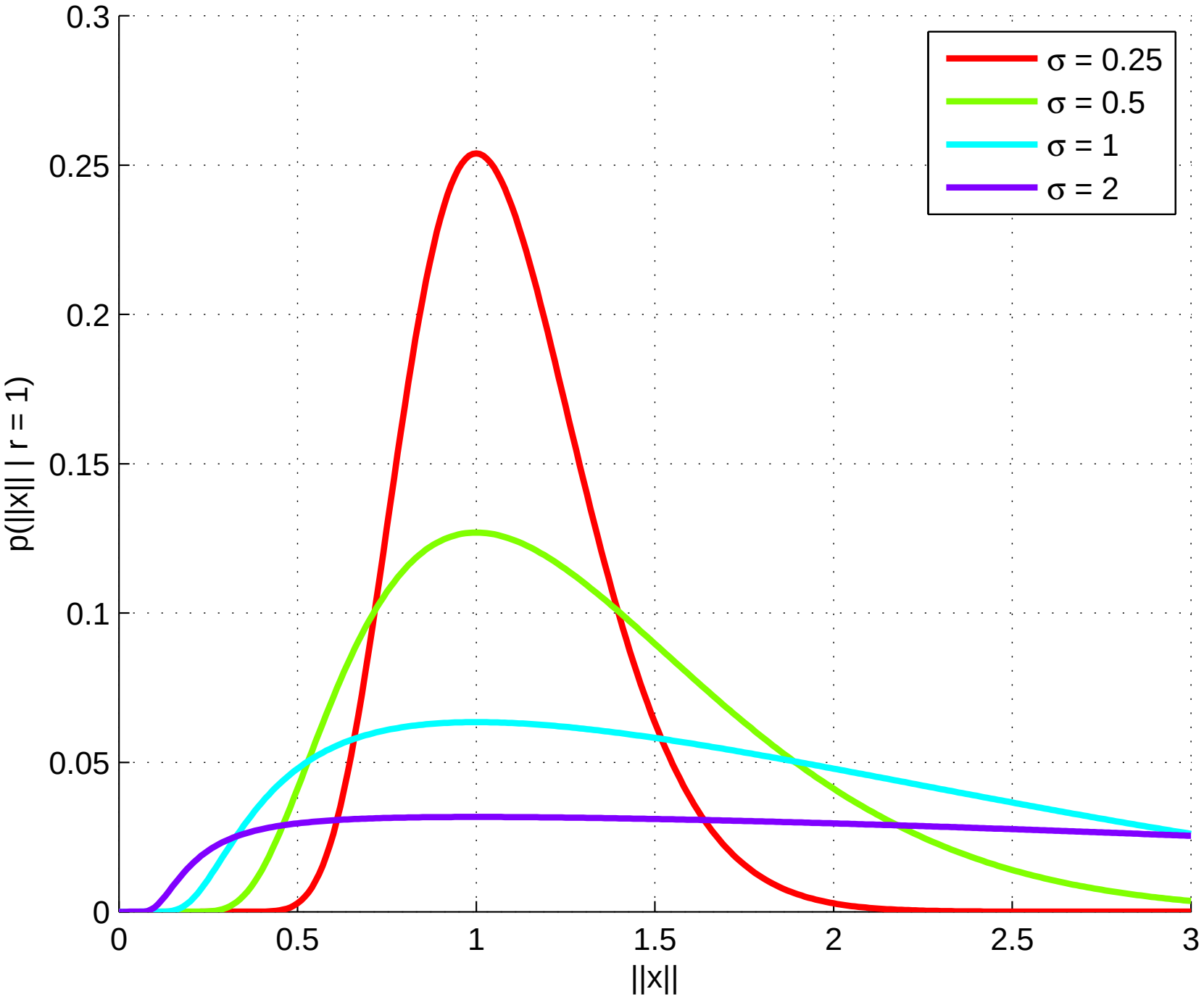
$$q(x \mid x_0) = \begin{cases} \mathrm{U}_{0,1}(x) & \text{(independent) uniform sampling} \\ \mathrm{Be}(x; \frac{x_0}{T} + 1, \frac{1-x_0}{T} + 1) & \text{'beta' diffusion (crawler)} \quad T \text{ – temperature} \\ p(x) & \text{(independent) Gibbs sampler} \end{cases}$$

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

Thank You


σ = 0.25
σ = 0.5
σ = 1
σ = 2
p(||x|| | r = 1)
||x||
0
0.05
0.1
0.15
0.2
0.25
0.3
0
0.5
1
1.5
2
2.5
3


σ = 0.25
σ = 0.5
σ = 1
σ = 2
p(||x|| | r = 1)
||x||
0
0.05
0.1
0.15
0.2
0.25
0.3
0
0.5
1
1.5
2
2.5
3


σ = 0.25
σ = 0.5
σ = 1
σ = 2
p(||x|| | r = 1)
||x||
0
0.05
0.1
0.15
0.2
0.25
0.3
0
0.5
1
1.5
2
2.5
3

OPPA European Social Fund
Prague & EU: We invest in your future.