Institut für Kartographie und Geoinformatik | Leibniz Universität Hannover

Simulation and Markov chain Monte Carlo

Claus.Brenner@ikg.uni-hannover.de

Leibniz Universität Hannover

Simulation and Markov chain Monte Carlo

- Motivation
  - Expectation and maxima of distributions
- Drawing from distributions
  - Simple methods: uniform distribution, inversion method
  - Accept-Reject
  - Importance Sampling
  - Sampling-Importance-Resampling
- Example: Art Gallery Problem
  - Gradient ascent
  - Uniform sampling
  - Markov chain Monte Carlo.

Motivation:
Expectations and maxima of distributions

Prior, likelihood and posterior

- Bayes theorem:

$$p(A \mid B) = \frac{p(B \mid A) \cdot p(A)}{p(B)} = \frac{p(B \mid A) \cdot p(A)}{\sum_{A'} p(B \mid A') \cdot p(A')}$$

$$\propto p(B \mid A) \cdot p(A)$$

- Typical application in machine learning:
  - M … model
  - D … data

$$p(M \mid D) \propto p(D \mid M) \cdot p(M) \qquad \left(= p(D, M) \right)$$

Prior, likelihood and posterior

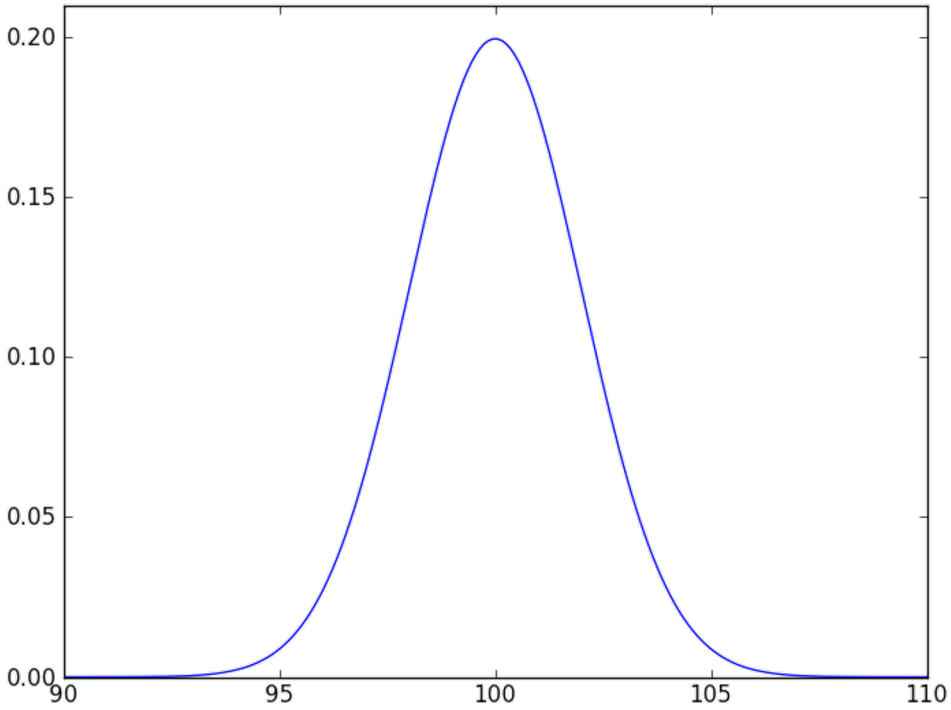
$$\underbrace{p(M \mid D)}_{\text{posterior}} \propto \underbrace{p(D \mid M)}_{\text{likelihood}} \cdot \underbrace{p(M)}_{\text{prior}}$$

- Prior: what we know **in advance** about our model
- Likelihood: how probable the data D are, given the model M
- Posterior: what we know **afterwards** (after using the data D) about our model.

Very simple example: prior

- A producer of "yardsticks" (100 cm yardsticks!) describes their deviation from the ideal length by a normal distribution ("production error").
- As long as we don't have additional information, our prior model is:

$$p(\mu) = \mathrm{N}(\mu \mid \mu_0, \sigma_0^2) \quad = p(\mathrm{model})$$



Example:

$$\mu_0 = 100$$

$$\sigma_0 = 2$$

Very simple example: likelihood

- Using a (reference) measurement instrument, we check the length of the yardstick
- The producer of the instrument specifies the measurement accuracy by a normal distribution ("measurement error"):

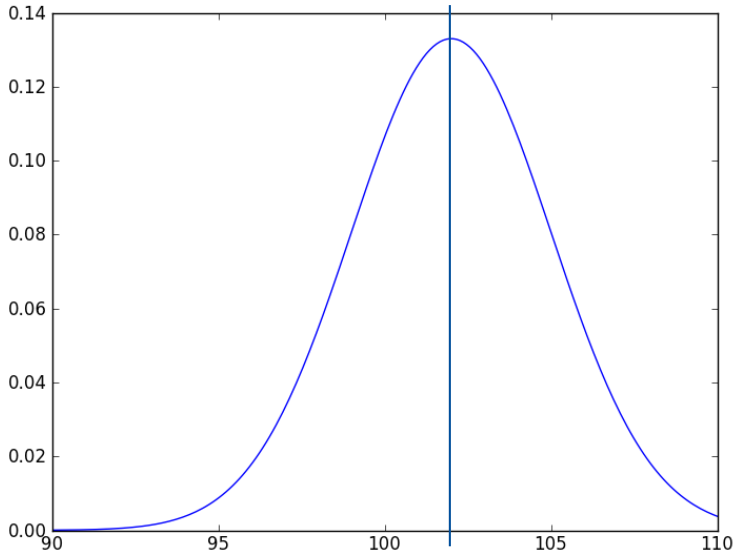
$$p(x) = \mathrm{N}(x \mid \mu, \sigma^2) \quad = p(\mathrm{data} \mid \mathrm{model})$$

- This tells us: if the true length is μ, then our measurements are distributed around μ with variance σ^2

Example:

$\mu = 102$

$\sigma = 3$

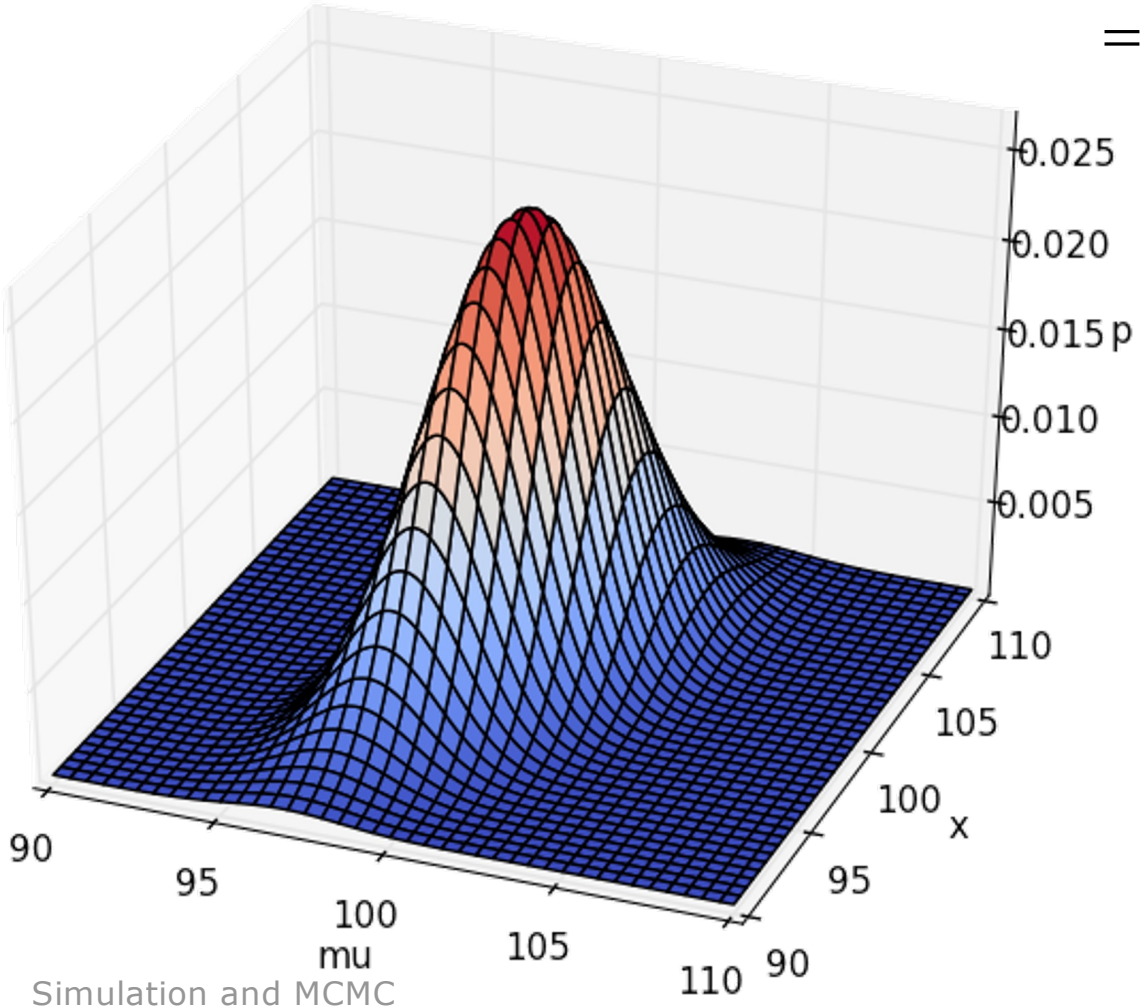


Very simple example: posterior

- The posterior is given by:

$$p(M \mid D) \propto p(D \mid M) \cdot p(M)$$

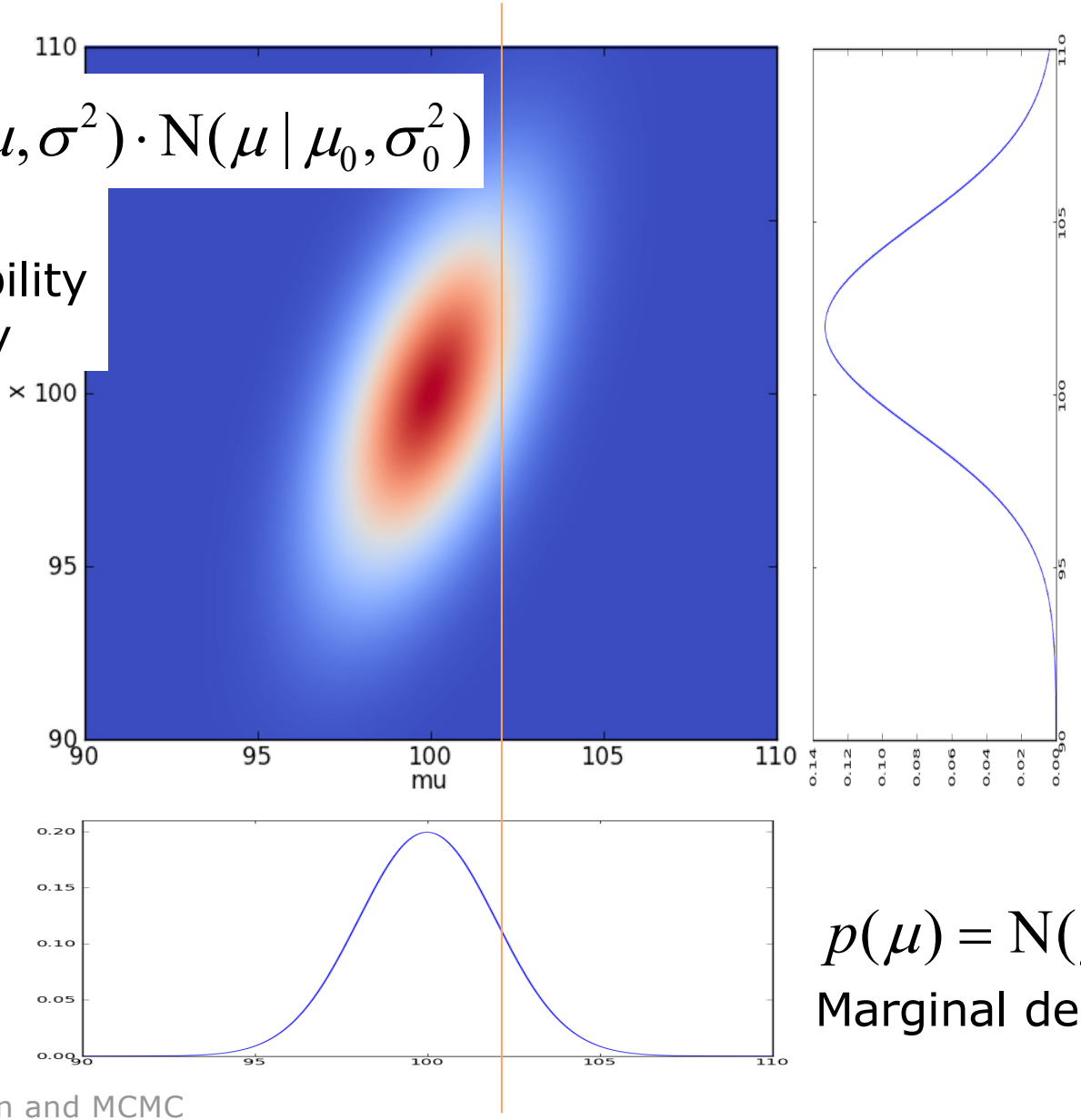
$$= \underbrace{\mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)}_{\text{Function of x and } \mu}$$



Very simple example: posterior

$$\mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)$$

Joint probability density



$$\mathrm{N}(x \mid \mu, \sigma^2)$$

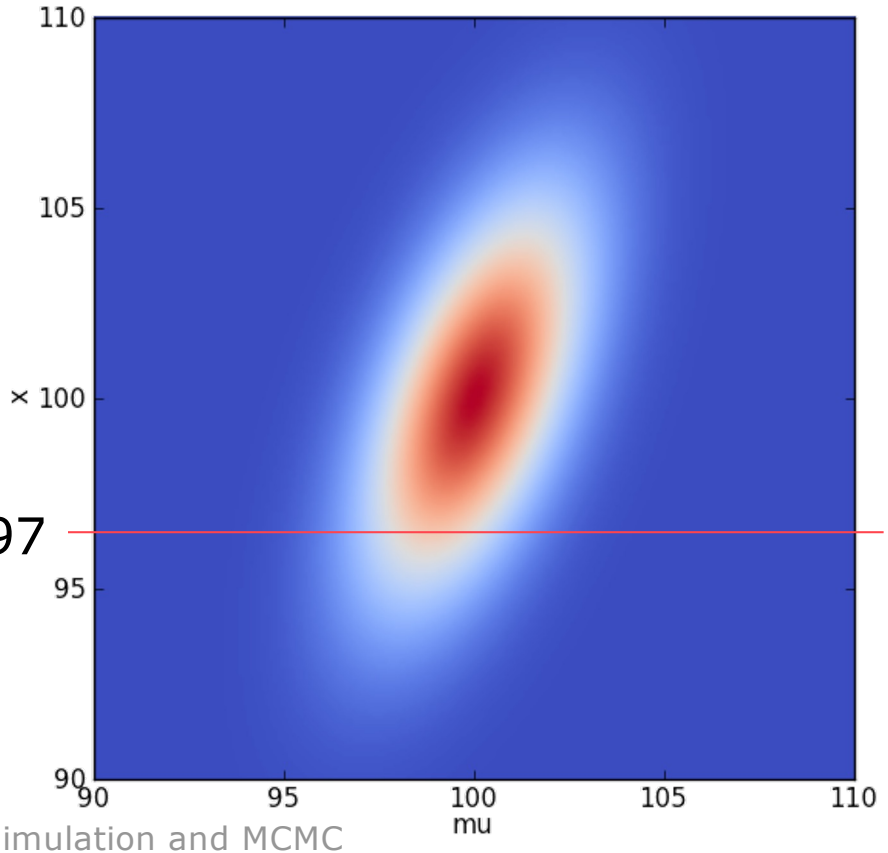
Conditional density

$$p(\mu) = \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)$$

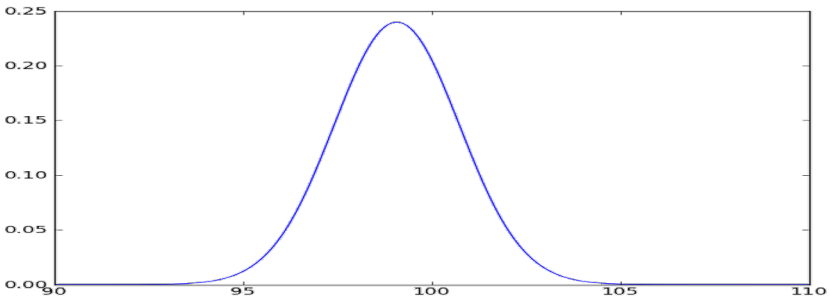
Marginal density

Very simple example: posterior

- We want to know:
  How does our knowledge about μ change after a measurement?
- We are interested in the "horizontal" conditional density!



$$p(\mu \mid x)$$



Very simple example: posterior

- Therefore, we have to solve the equation for μ

$$p(M \mid D) = p(\mu \mid x) \propto \mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)$$

Sought for: function of μ for a fixed, given x

- Solution (by "completing the square"):

$$p(M \mid D) = p(\mu \mid x) = \mathrm{N}(\mu \mid \mu_{\mathrm{post}}, \sigma^2_{\mathrm{post}})$$

  - The posterior distribution is (again) a normal distribution
  - Mean and variance can be computed easily from the parameters of the prior distribution and the likelihood.

Very simple example: posterior

- Parameters of the posterior:
  - It is normal distributed:

$$p(M \mid D) = p(\mu \mid x) = \mathrm{N}(\mu \mid \mu_{\mathrm{post}}, \sigma^2_{\mathrm{post}})$$

  - with these parameters:

$$\mu_{\mathrm{post}} = \frac{x/\sigma^2 + \mu_0/\sigma_0^2}{1/\sigma^2 + 1/\sigma_0^2}$$

$$1/\sigma^2_{\mathrm{post}} = 1/\sigma^2 + 1/\sigma_0^2$$

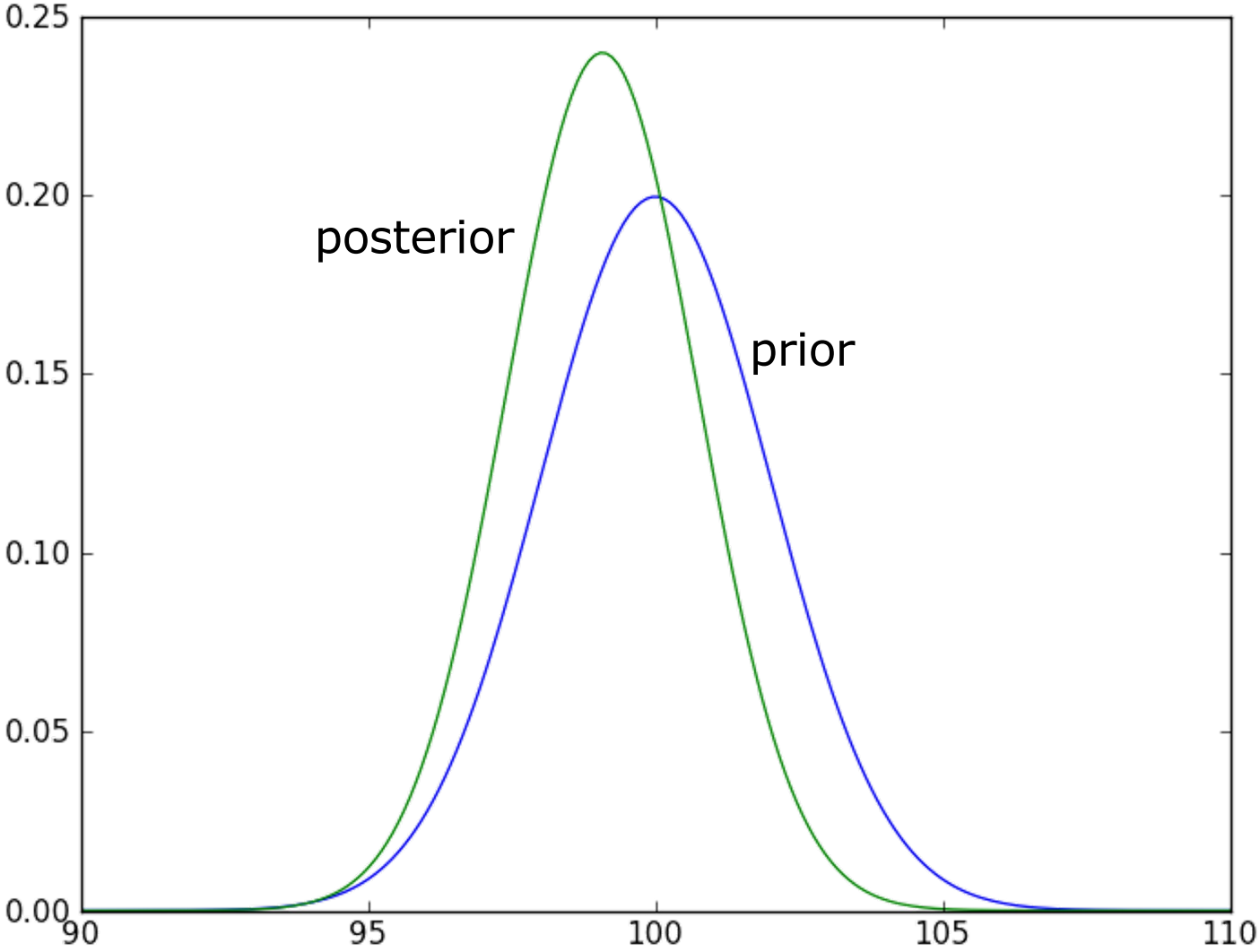
(weighted mean)

  - which means for the values we assumed:

$$\mu_{\mathrm{post}} = 99.0769$$

$$\sigma_{\mathrm{post}} = 1.66410$$

Very simple example: posterior



Problem: this trick only works in (very) special cases

- In our case, we were able to compute the posterior distribution

$$p(\mu \mid x) = \frac{\mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)}{\int \mathrm{N}(x \mid \mu', \sigma^2) \cdot \mathrm{N}(\mu' \mid \mu_0, \sigma_0^2) d\mu'}$$

- Reason why this was so easy: we have chosen a suitable prior distribution (the normal distribution)
  - Due to this, our posterior distribution is also a normal distribution, for which mean and variance can be computed easily. We need not compute the normalization factor (denominator) explicitly
  - Such a choice is termed "**conjugate prior**"
  - Thus, a conjugate prior is a functional form, which leads, when multiplied with the likelihood, to the same functional form (but with different parameters).

General case

- In general, the combination of prior and likelihood leads to a function which cannot be given analytically
  - I.e. in general, we are unable to write the function in terms of a "simple function in μ" (as in the example)
- However, we are able to compute the (numeric) value of the function for any given variables
- In general, we are interested in computing
  - moments: mean, variance, …

$$\int f(M)\,p(M \mid D)\,dM$$

In our example: $\mu_{\text{post}}, \sigma_{\text{post}}$

  - maxima:

$$M^* = \arg\max_M p(M \mid D)$$

Maximum a posteriori (MAP) estimation

General approach

- We "know" a distribution, given by its density

$$p(x)$$

- From this, we want to compute certain quantities, e.g.:
  - Expectation of a function f:

$$\mathrm{E}[f] = \int f(x)\,p(x)\,dx$$

(e.g. f(x)=x, f(x)=x^2, …)

  - Maximum:

$$x^* = \arg\max_x p(x)$$

- We cannot give these quantities in analytical form
- But we are able to compute $p(x)$ for any given x.

Computation of expected values using sampling

- We want to compute the expectation of a function *f* under a given distribution (here: density) *p*

$$\mathrm{E}[f] = \int f(x)\,p(x)\,dx$$

- Idea: if we are able to draw samples from the distribution *p*, then the expectation can be approximated by:

$$\mathrm{E}[f] \approx \frac{1}{m}\sum_{j=1}^{m} f(x_j) \qquad \text{with } x_j \sim p(x)$$

How good is this approximation?

$$\hat{f} := \frac{1}{m}\sum_{j=1}^{m} f(x_j) \quad \approx \mathrm{E}[f]$$

- Expectation of the estimator:

$$\mathrm{E}[\hat{f}] = \mathrm{E}[\frac{1}{m}\sum_{j=1}^{m} f(X_j)] = \frac{1}{m}\sum_{j=1}^{m} \mathrm{E}[f(X_j)] = \frac{1}{m} \cdot m \cdot \mathrm{E}[f] = \mathrm{E}[f]$$

- Variance of the estimator:

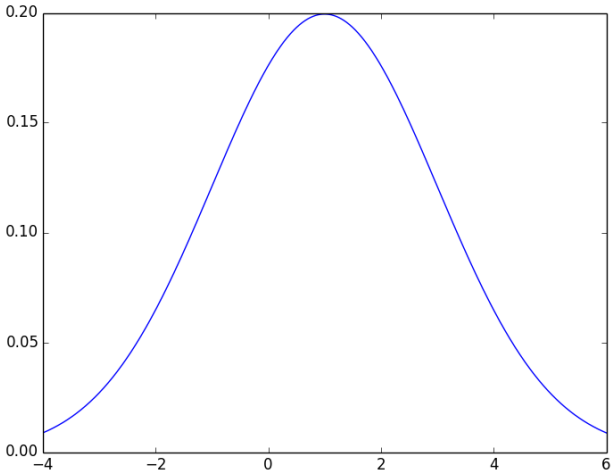
$$\mathrm{var}[\hat{f}] = \mathrm{E}\left[(\hat{f} - \mathrm{E}[\hat{f}])^2\right] = \frac{1}{m} \cdot \mathrm{E}\left[(f - \mathrm{E}[f])^2\right]$$

(see Bishop, exercise 11.1)

Example: mean of the normal distribution

- Original distribution: normal distribution

$$\mu = 1, \sigma^2 = \mathrm{var}[x] = 4$$



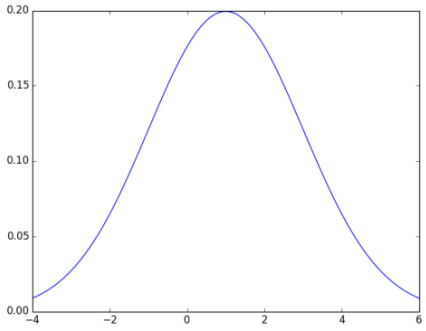
- Computation of the mean value

$$f(x) := x, \quad \text{i.e.} \quad \mathrm{E}[f] = \int x \cdot p(x) dx$$

- Thus, the estimator is simply:

$$\mathrm{E}[f] \approx \hat{f} := \frac{1}{m} \sum_{j=1}^{m} x_j \qquad \text{with } x_j \sim \mathrm{N}(x \mid \mu, \sigma^2)$$

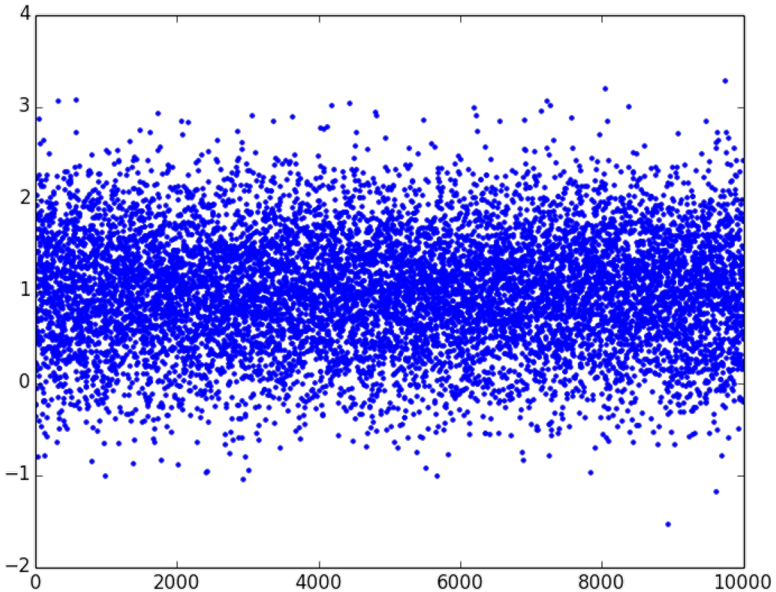
Example: mean of the normal distribution



Original

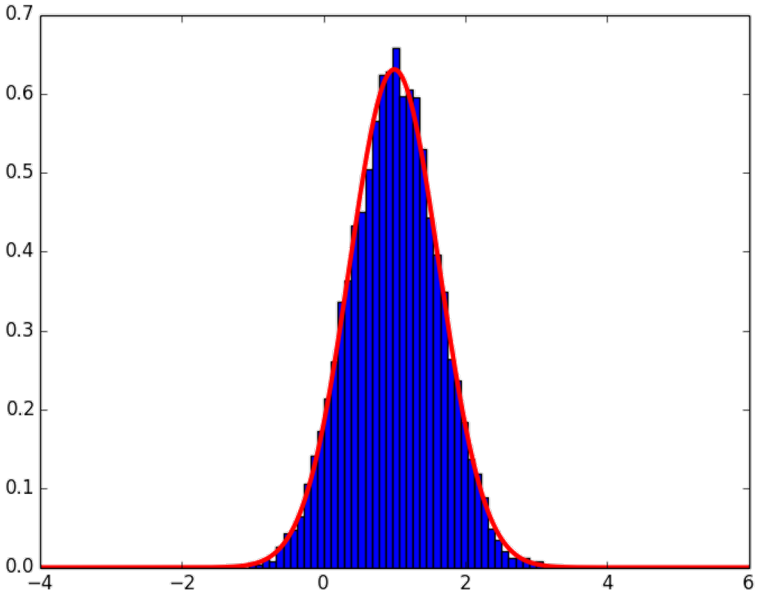
$\mu = 1,$

$\mathrm{var}[x] = 4$



10.000 computations of

$$\hat{f} := \frac{1}{10}\sum_{j=1}^{10} x_j$$



Histogram of 10.000 estimated mean values. Red curve is a normal distribution with:

$$\mu = 1,\ \mathrm{var}[x] = \frac{4}{10}$$

Sampling from distributions

Sampling from distributions

- Our task is therefore: we want to sample from a (complex) distribution

$$x_j \sim p(x)$$

(In order to compute: $\mathrm{E}[f] \approx \frac{1}{m} \sum_{j=1}^{m} f(x_j))$

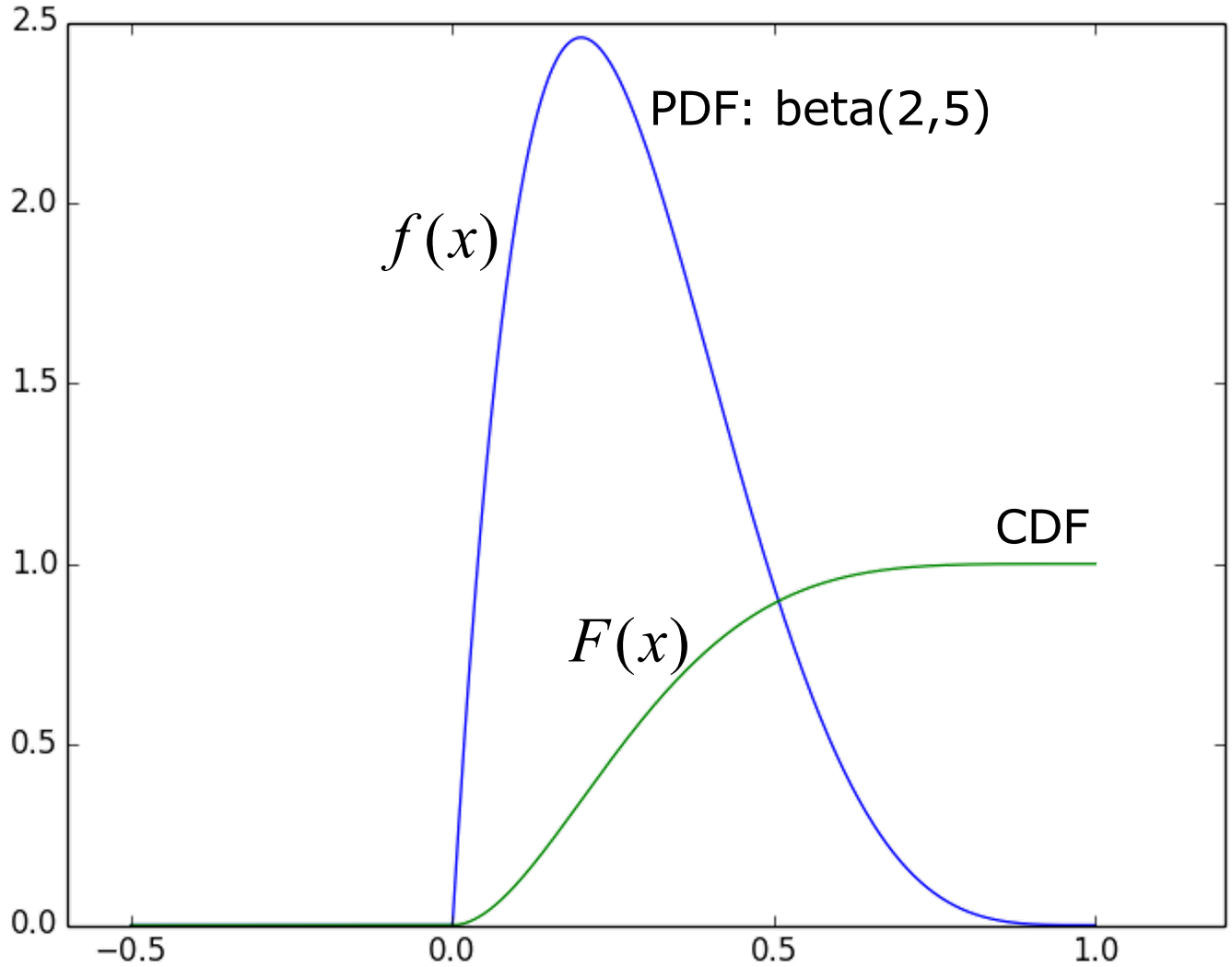
- = Random sample generator for a random variable with density $p(x)$

Simple distributions: uniform distribution

- Uniform distribution: standard "random generators"
  - "randomness" is not possible on deterministic computers
  - → pseudo random number generators
  - Transformation D, sequence of random numbers:
    - u_0, $u_1 = Du_0$, $u_2 = Du_1 = D^2u_0$, $u_3 = Du_2$…
  - The algorithms compute a sequence of integers in [0, max) which appear to be independent, identically distributed samples from a uniform distribution
    - There are collections of programs which test this property (so-called "die hard" tests (https://en.wikipedia.org/wiki/Diehard_tests)
    - This approach is not free of problems (since it is not truly random – it just fulfils the tests)
  - Uniform random number generators are available in standard programming environments.

Simple distributions: “inversion trick”

- Density (PDF) and cumulative density function (CDF)



Simple distributions: "inversion trick"

- Let U be uniformly distributed

$$U \sim \text{uniform}(0,1)$$

- Then define:

$$V := F^{-1}(U) \qquad \text{(Inverse of the CDF)}$$

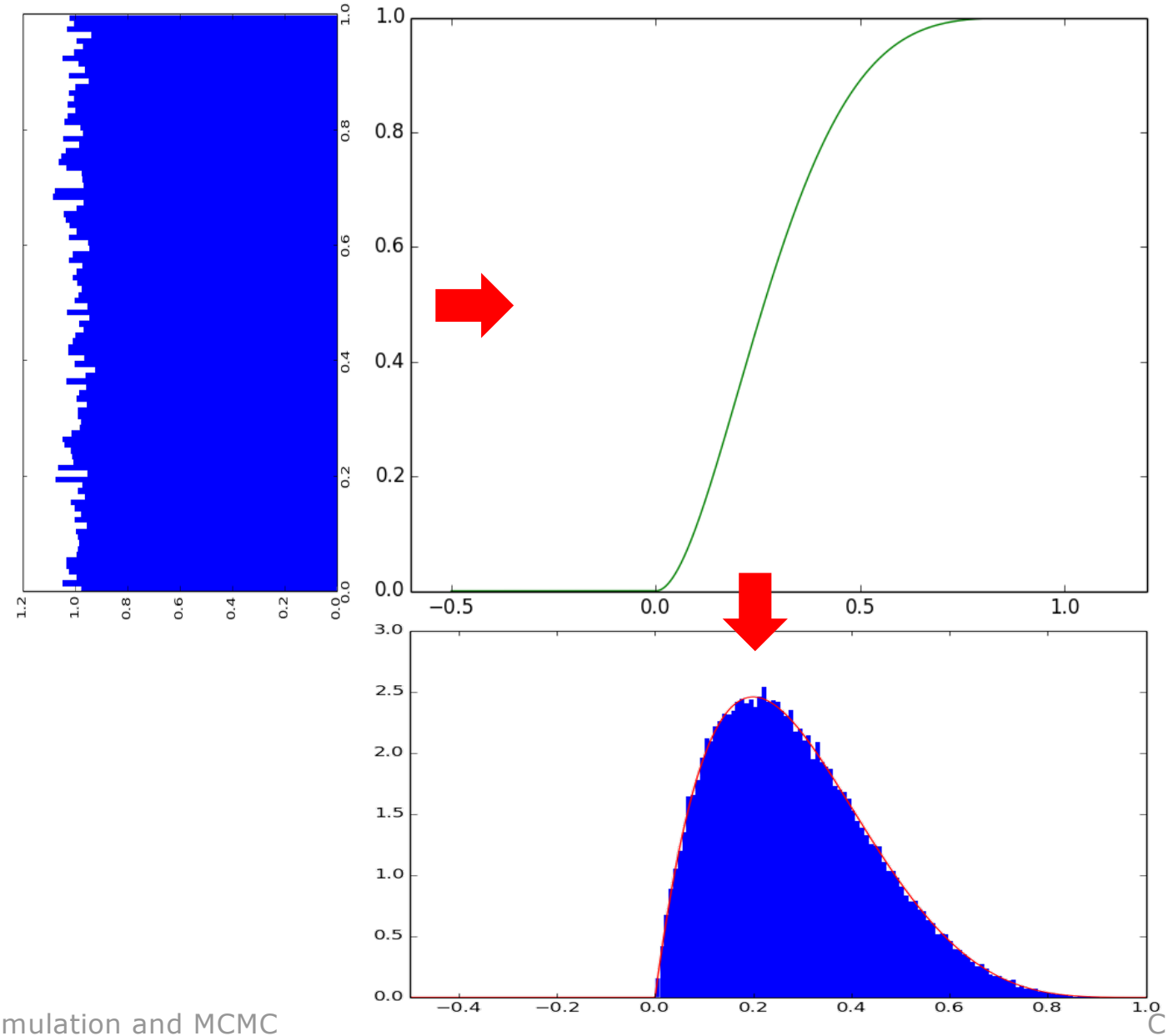
- Then, it follows for the CDF of V, F_V:

$$F_V(x) = P(V \leq x) = P(F^{-1}(U) \leq x) = P\big(U \leq F(x)\big) = F(x)$$

- I.e., if U is distributed uniformly on [0,1], then $F^{-1}(U)$ has the CDF $F(x)$
- To sample $X \sim f(x)$ one can draw $U \sim \text{uniform}(0,1)$ and then compute $x = F^{-1}(u)$
- In the general case: generalized inverse

$$F^{-}(u) := \inf\{x \mid F(x) \geq u\}$$

Simple distributions: “inversion trick”



Simple distributions: “inversion trick”

- “Inversion trick” works only if the inverse CDF $x = F^{-1}(u)$ can be computed (and in an efficient manner)
- → not suitable for distributions which are not given in a usable form or for which an inverse CDF cannot be computed or cannot be computed efficiently.

The accept-reject method (rejection sampling)

- Theorem: The simulation of

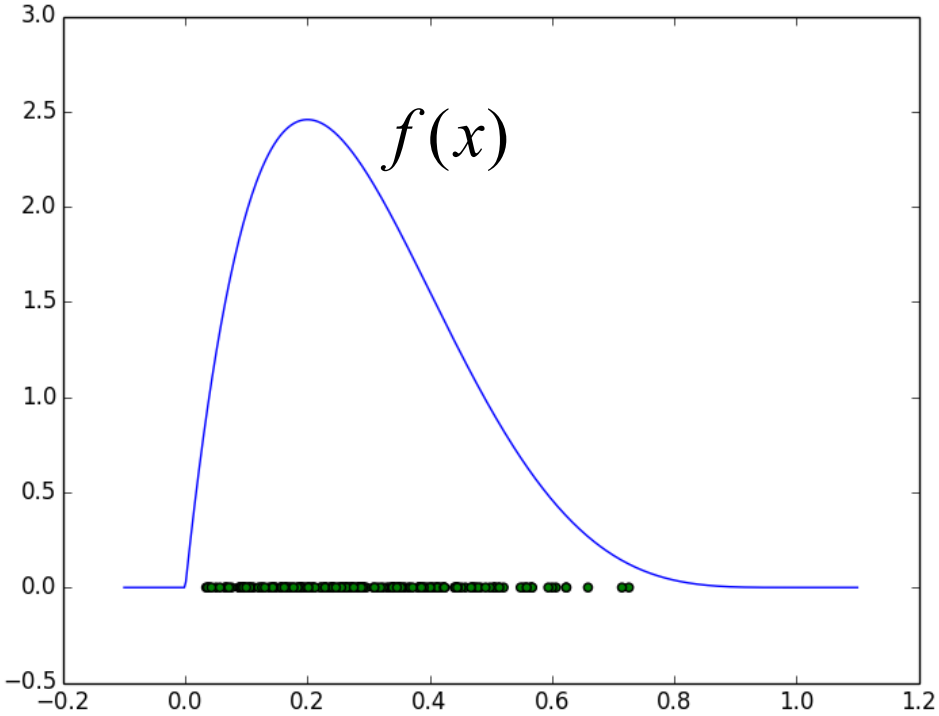
$$X \sim f(x)$$

and

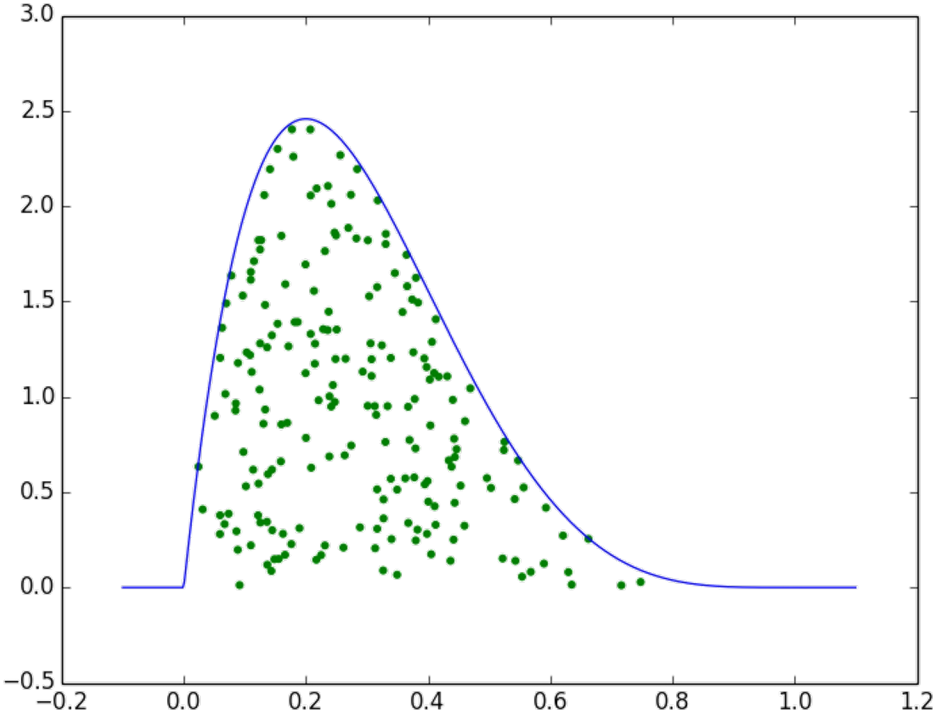
$$(X,U) \sim \text{uniform}\{(x,u) \mid 0 < u < f(x)\}$$

are equivalent.

Visualization



$$X \sim f(x)$$

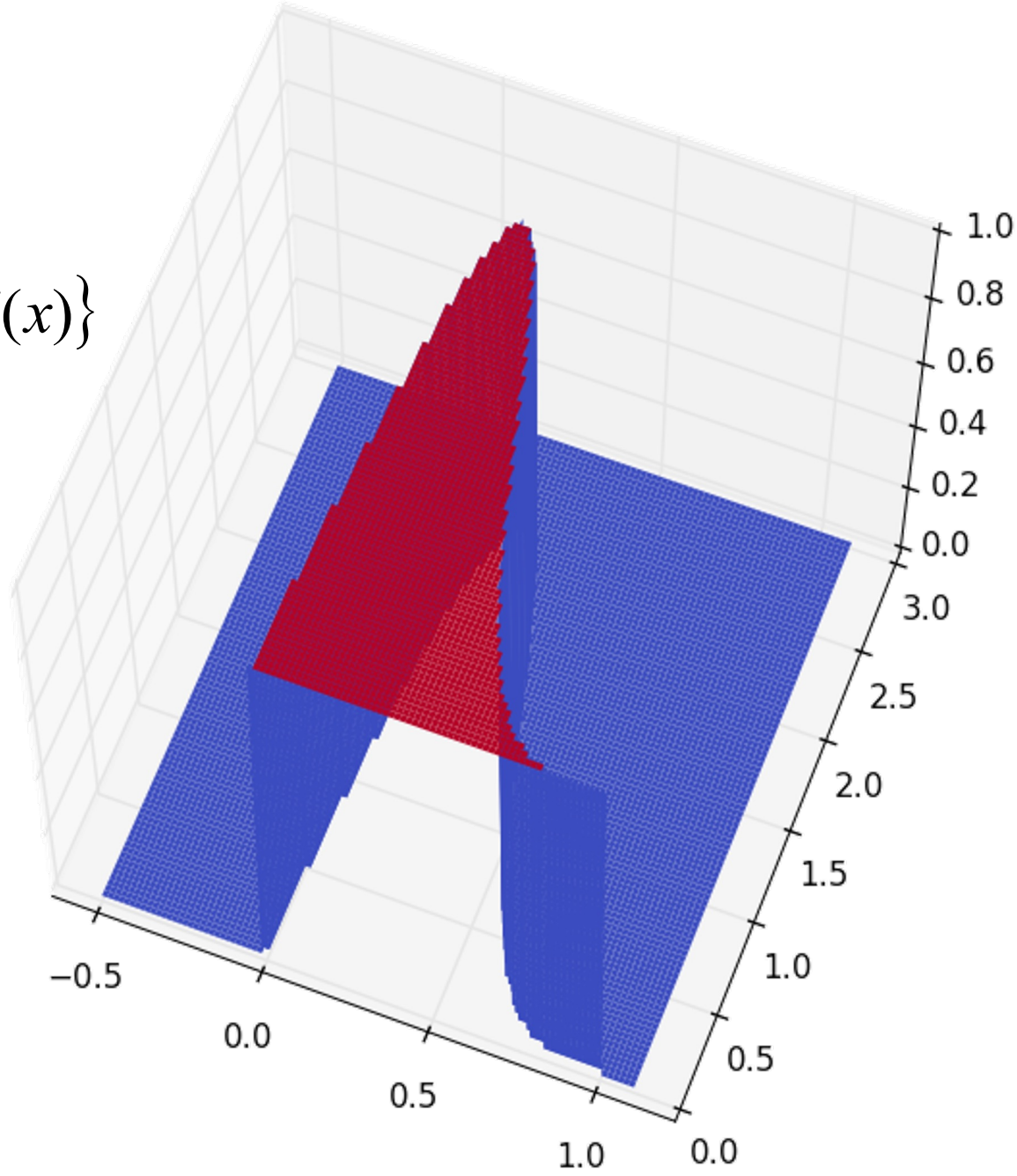


$$(X,U) \sim \text{uniform}\{(x,u) \mid 0 < u < f(x)\}$$

Visualization

- "Replacement" density:

$$\text{uniform}\{(x,u) \mid 0 < u < f(x)\}$$



Accept-reject

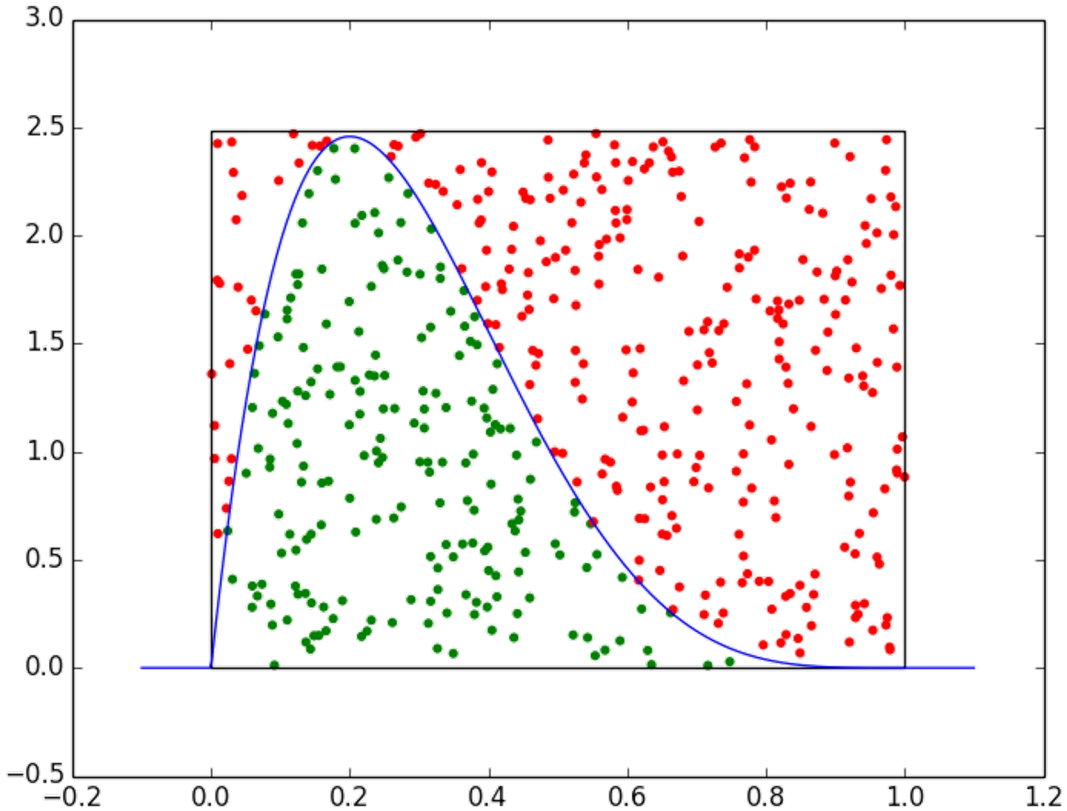
- What have we gained by this?
- The distribution $\text{uniform}\{(x,u) \mid 0 < u < f(x)\}$ can be obtained easily by using a simple distribution
- Example:
  - For an arbitrary distribution, which is defined on [a, b] and which is zero outside, and assumes a maximum value of m, one can sample as follows:

Accept-reject (special case)

1. Draw $X \sim \text{uniform}[a,b]$
2. Draw $U \sim \text{uniform}[0,m]$
3. Accept, if $0 \leq u \leq f(x)$

Accept-reject

- Uniform distribution on $[a,b]\times[0,m]$
- The accepted subset (green) is uniformly distributed on $\{(x,u)\,|\,0<u<f(x)\}$



Accept-reject: properties

- Efficiency depends on the probability of acceptance
  - Area which is used for the uniform simulation:

$$[a,b]\times[0,m] \qquad A_1 = (b-a)\cdot m$$

  - Area which is accepted (*f* is the density):

$$A_2 = \int_a^b f(x)dx = 1$$

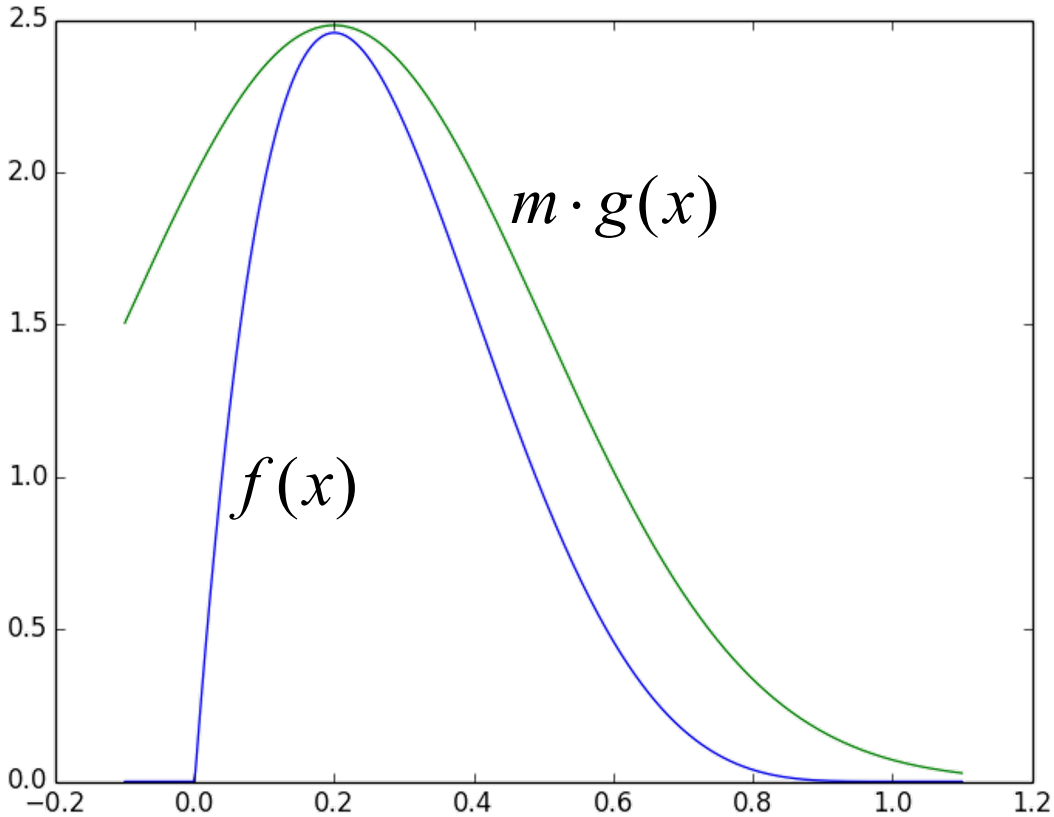
  - Acceptance probability:

$$\frac{A_2}{A_1} = \frac{1}{(b-a)\cdot m}$$

(In our example: 0.4)

Accept-reject: working with instrumental densities

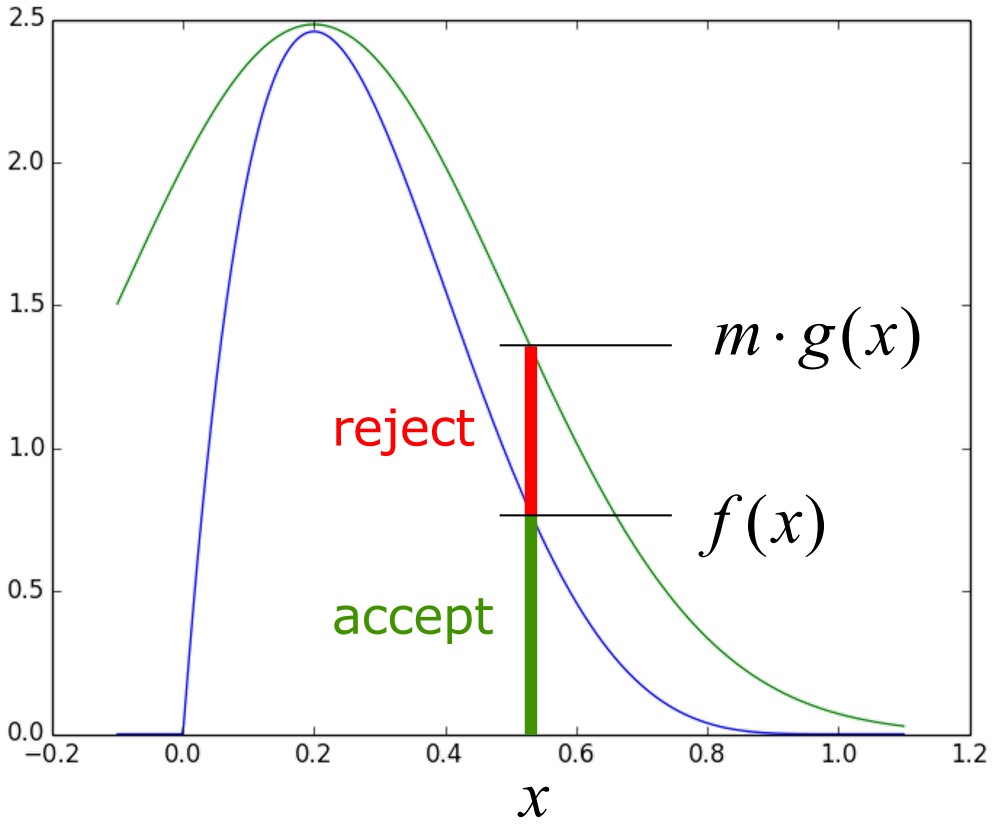
- Instead of a uniform density, one uses an **instrumental density** *g(x)*
- This is multiplied with a factor of *m*, so that it is larger than or equal to the target density *f(x)*.



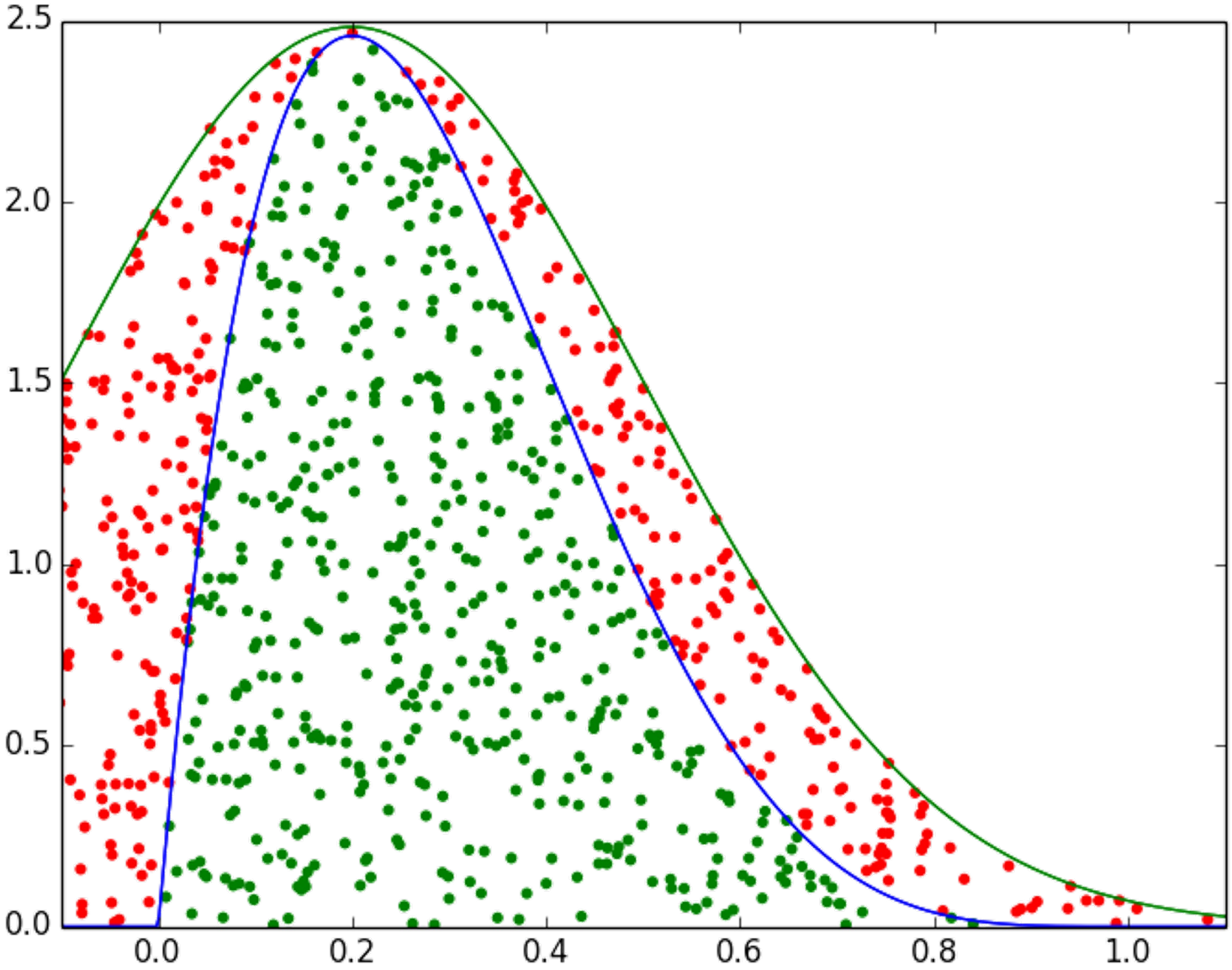
Accept-reject: the general case (using instrumental density *g*)

Accept-reject (general case)

1. Draw $X \sim g(x)$
2. Draw $U \sim \text{uniform}[0, m \cdot g(x)]$
3. Accept, if $0 \leq u \leq f(x)$



Accept-reject: the general case



Accept-reject: the general case

- Again, compute the acceptance probability
  - Area which is used for the uniform simulation:

$$A_1 = \int_{-\infty}^{+\infty} m \cdot g(x)dx = m$$

(Since *g* is a density function.)

  - Area which is accepted:

$$A_2 = \int_{a}^{b} f(x)dx = 1$$

  - Acceptance probability:

$$\frac{A_2}{A_1} = \frac{1}{m}$$

(In our example: 0.54, better than in the case of the uniform distribution)

Accept-reject: the general case

- Makes no difference if *f* is normalized or not
  - Useful when simulating $p(M \mid D) \propto p(D \mid M) \cdot p(M)$ since $p(D) = \int p(D \mid M) \cdot p(M) dM$ needs not to be computed
- Problem: how to find *m*?
- The instrumental density has to be larger than the density to be simulated, also for values $\pm\infty$
  - E.g.: the normal distribution cannot be used as an instrumental density if the target is a Cauchy distribution
  - …but the opposite case will work!
- For log-concave functions, the algorithm can be extended so that the instrumental density is iteratively adjusted and adapted to the density of *f*. (See Robert & Casella.)

Importance sampling

- Useful for the case where one wants to compute an expected value
- As we have seen above:

$$\mathrm{E}[f] = \int f(x)\,p(x)\,dx$$

  - Is approximated by:

$$\mathrm{E}[f] \approx \frac{1}{m}\sum_{j=1}^{m} f(x_j) \quad \text{with} \quad x_j \sim p(x)$$

- Problem: drawing of samples $\sim p(x)$ cannot be achieved easily.

Importance sampling

- Now: use an instrumental density *q(x)* in the integral:

$$\begin{aligned} \mathrm{E}[f] &= \int f(x)p(x)dx \\ &= \int f(x)\frac{p(x)}{q(x)}q(x)dx \\ &= \int \underbrace{\left(f(x)\frac{p(x)}{q(x)} \right)}_{\text{with } x_j \sim q(x)} q(x)dx \end{aligned}$$

- This is approximated by:

$$\mathrm{E}[f] \approx \frac{1}{m}\sum_{j=1}^{m} \overbrace{f(x_j)\frac{p(x_j)}{q(x_j)}}$$

Importance sampling

- This means:
  - Instead of drawing from our target density *p(x)*, we can draw from our instrumental density *q(x)*
  - In return, we have to correct each term by the weight *p(x)/q(x)*
  - This weight ("importance") gave the method its name

$$\mathrm{E}[f] \approx \frac{1}{m} \sum_{j=1}^{m} f(x_j) \underbrace{\frac{p(x_j)}{q(x_j)}}_{=: w_j}$$

Weight ("correction")

$$\text{with } x_j \sim \underbrace{q(x)}$$

Instrumental density

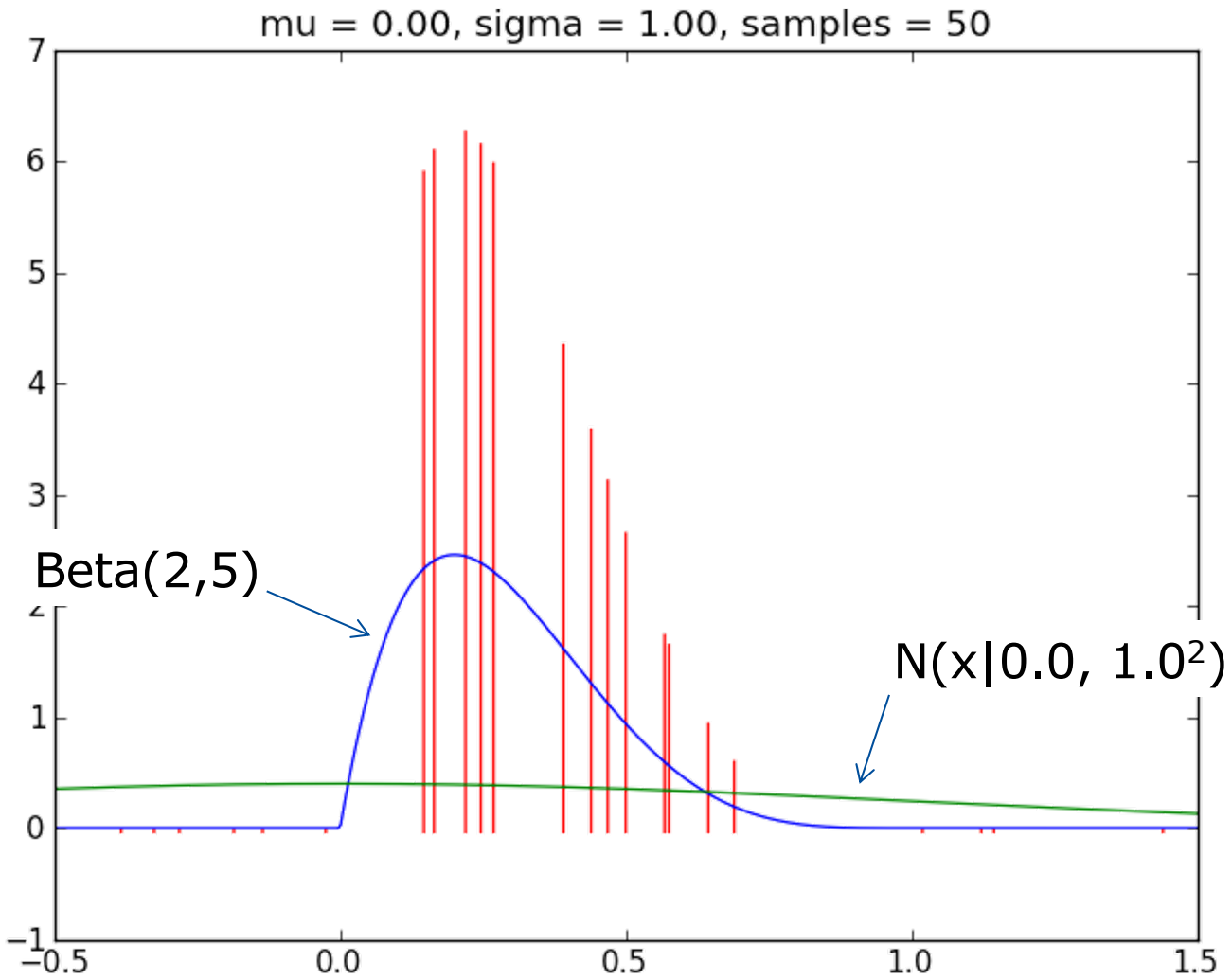
Example: importance sampling

- Beta(2, 5) distribution
- Instrumental density: taken from normal distribution
- Goal: computation of mean value:

$$\mathrm{E}[x] \approx \frac{1}{m}\sum_{j=1}^{m} x_j \frac{p(x_j)}{q(x_j)} = \frac{1}{m}\sum_{j=1}^{m} x_j w_j$$

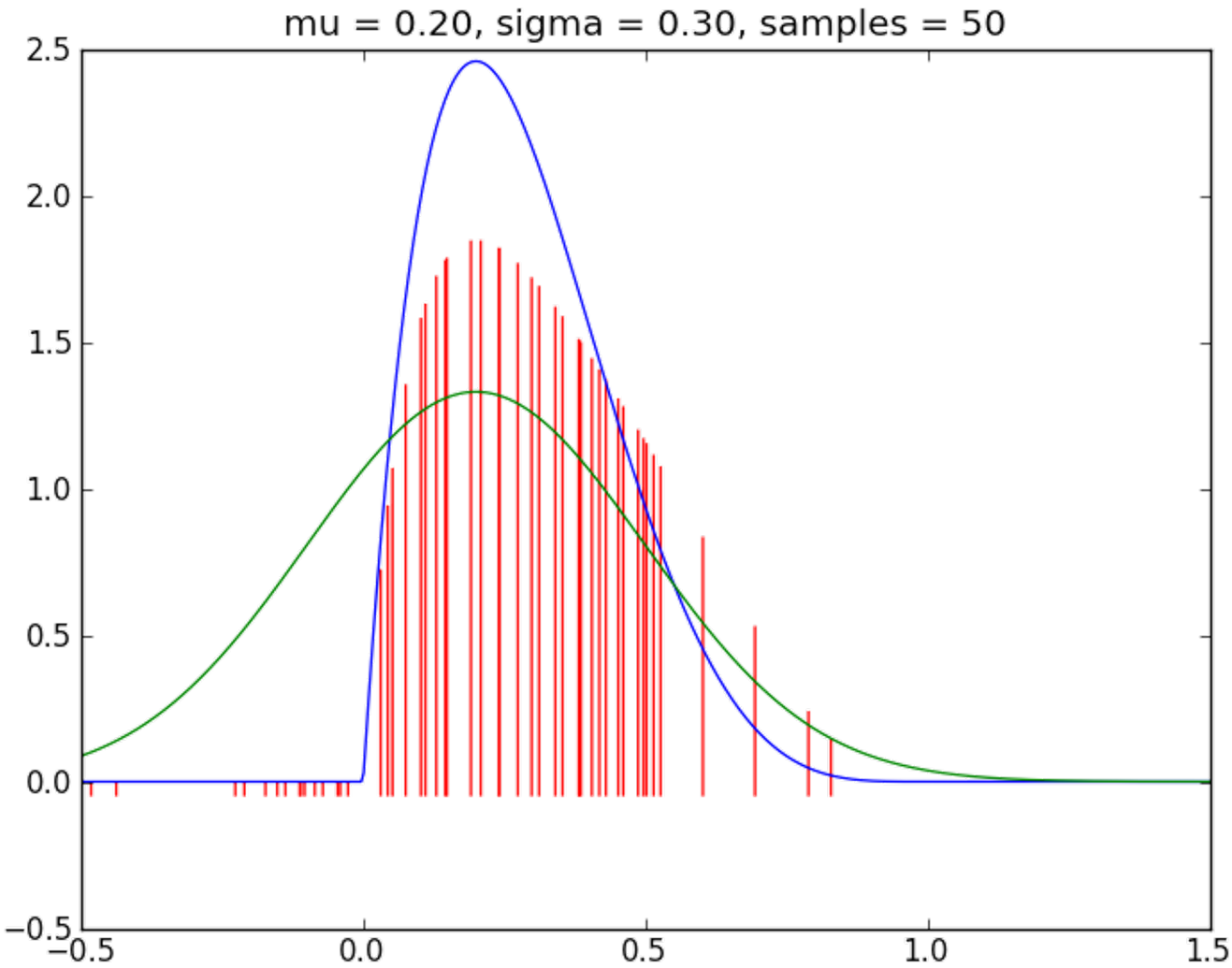
Example: importance sampling

- $q(x) = N(x|0.0, 1.0^2)$



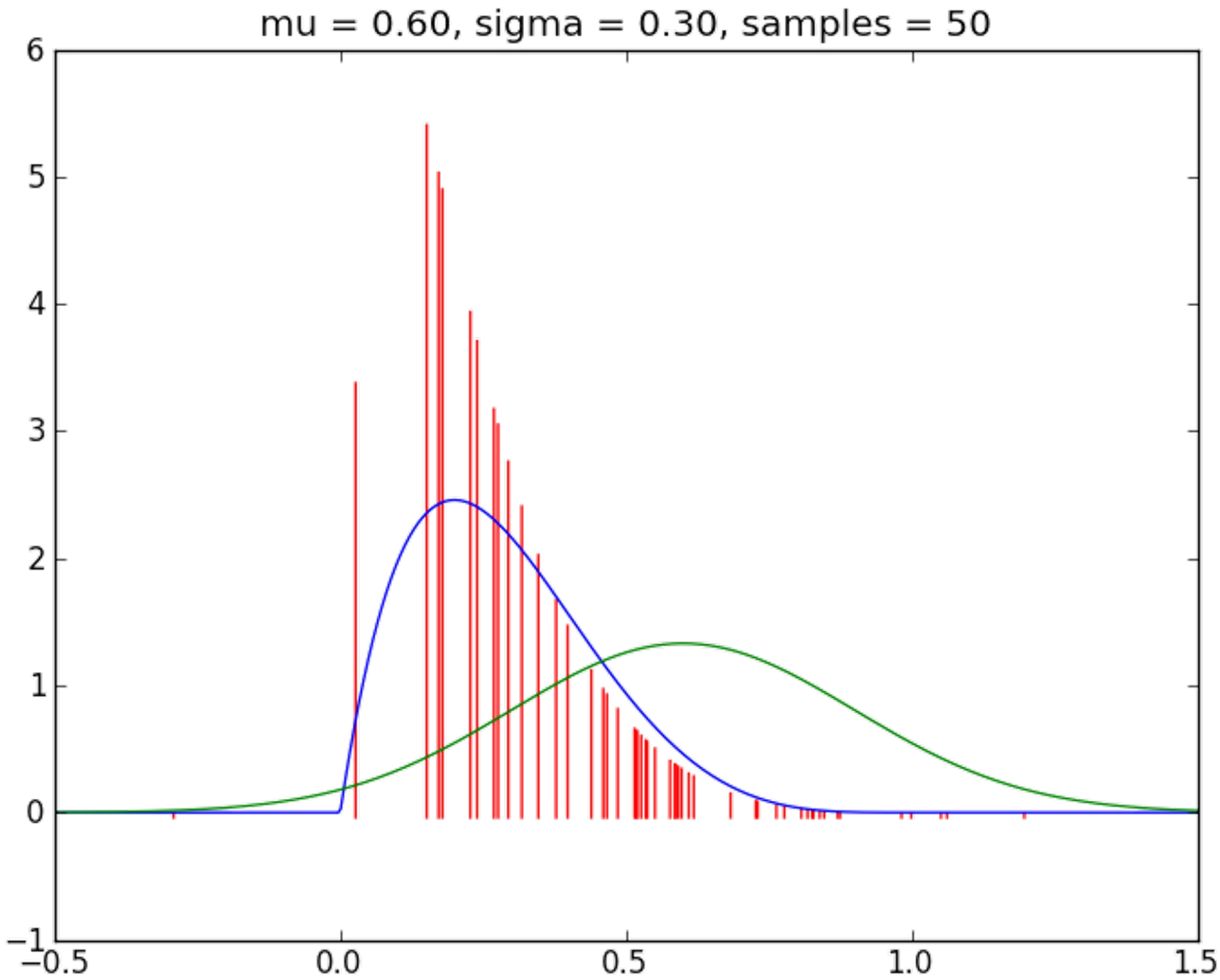
Example: importance sampling

- $q(x) = N(x|0.2, 0.3^2)$



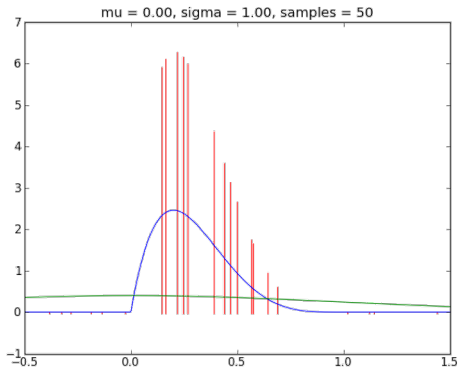
Example: importance sampling

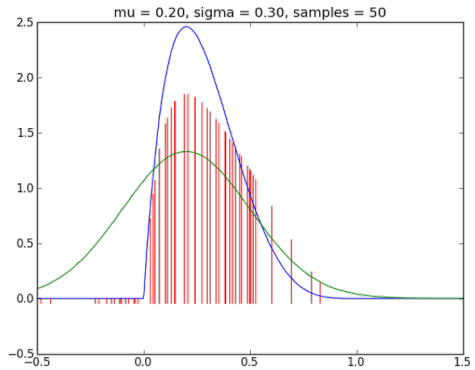
- $q(x) = N(x|0.6, 0.3^2)$

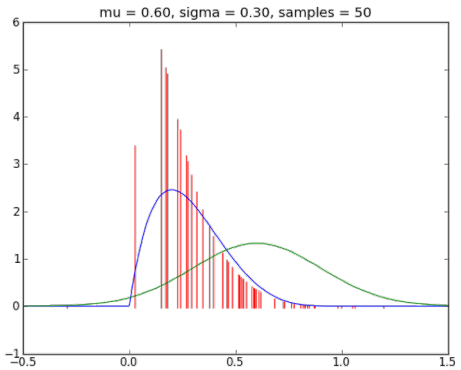


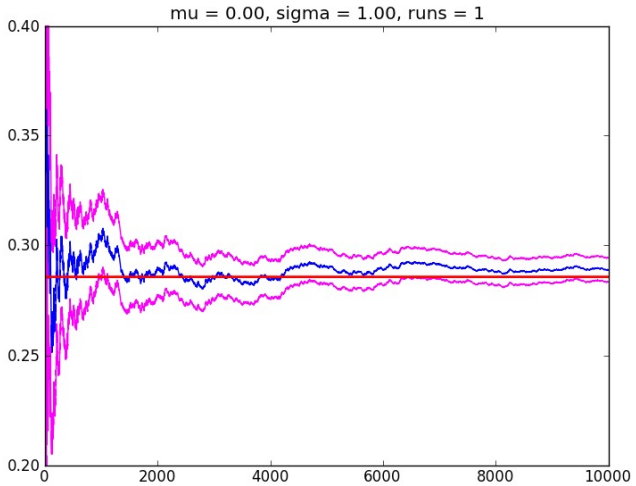
Importance sampling: convergence

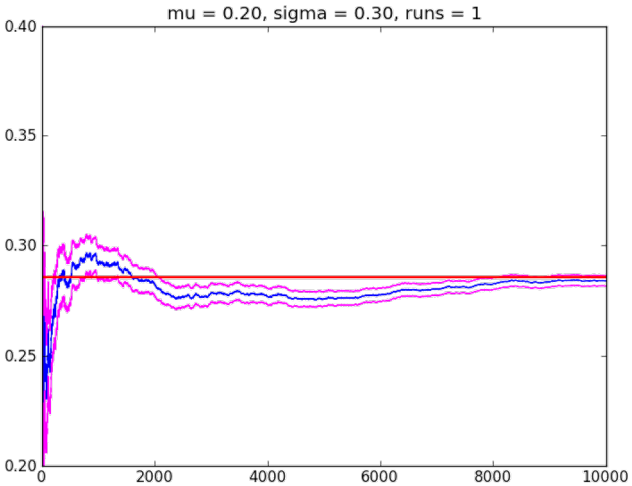
(correct mean value : $\mu_0 = 0.2857$)

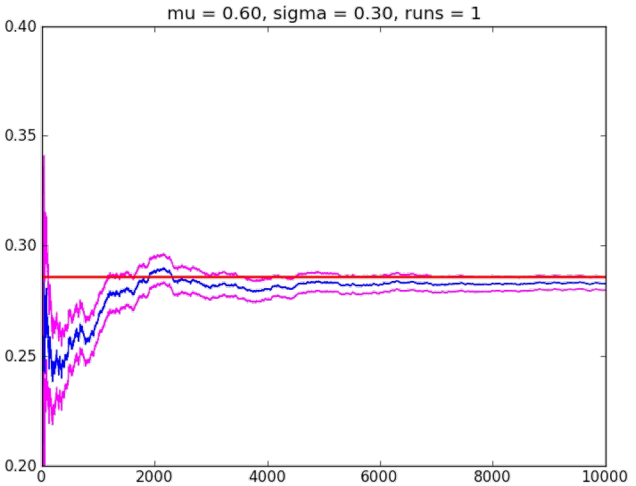






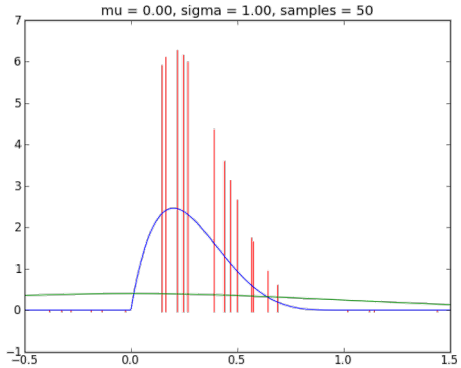


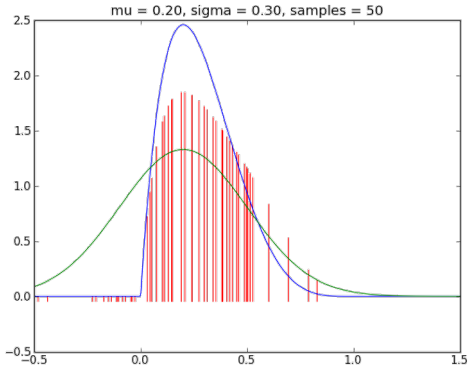


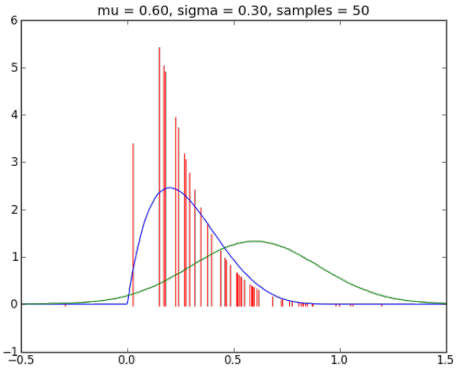


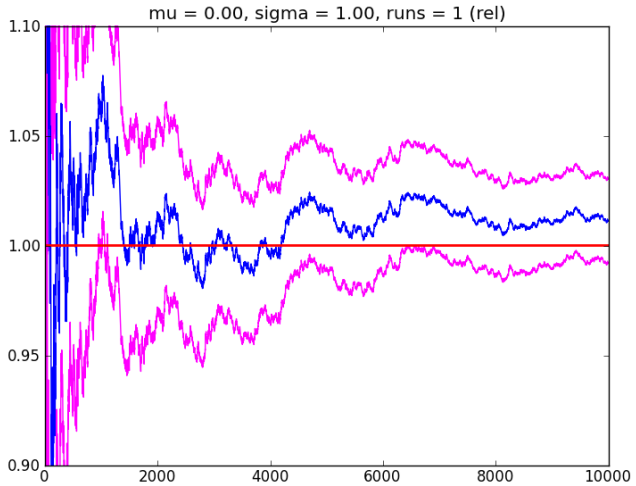
$$\frac{1}{m}\sum_{j=1}^{m} x_j w_j \quad \text{for } 0 < m \leq 10000$$

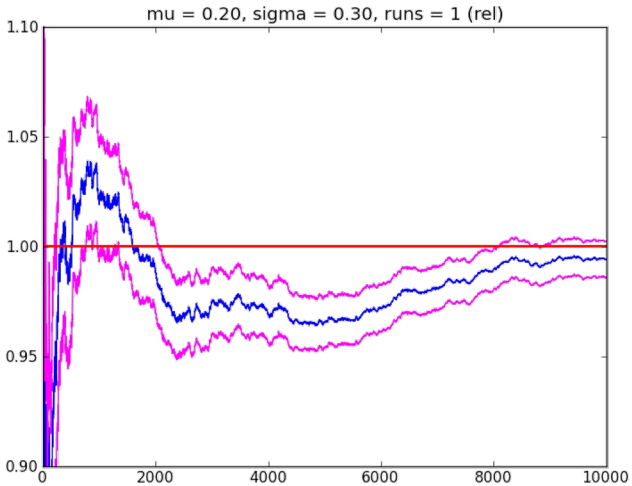
Importance sampling: convergence (relative deviation)

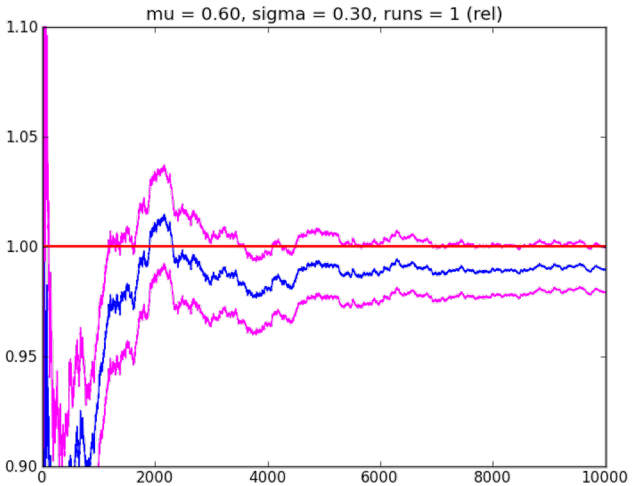






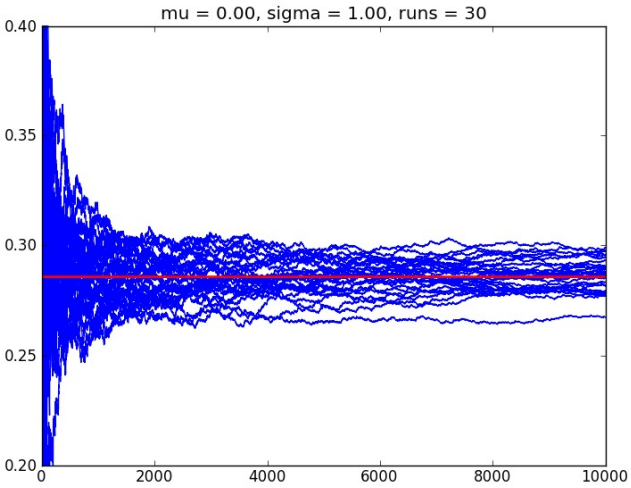


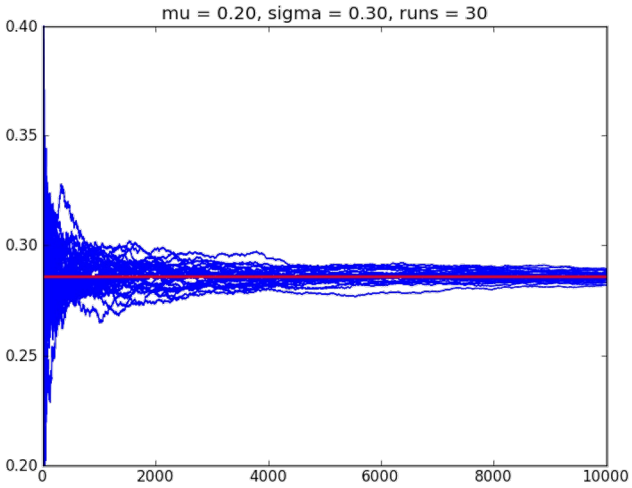


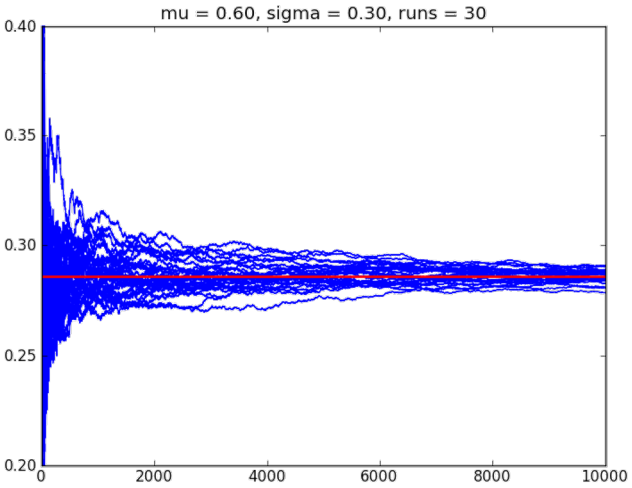


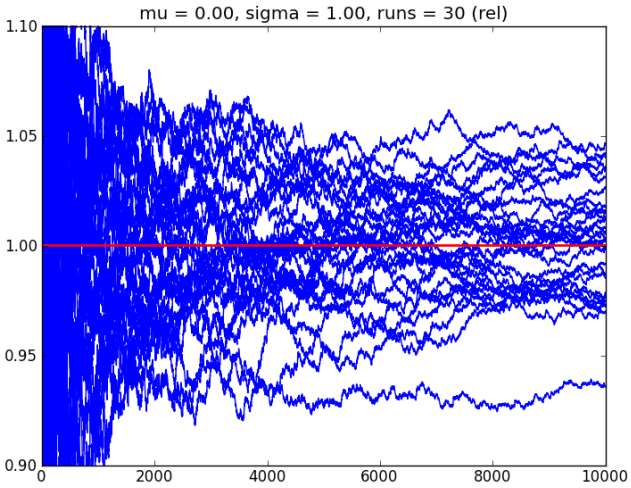
$$\frac{1}{m}\sum_{j=1}^{m} x_j w_j \Big/ \mu_0 \quad \text{for } 0 < m \leq 10000$$

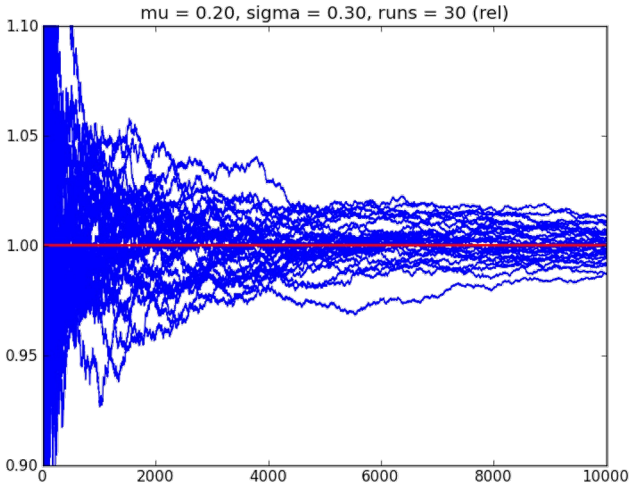
Importance sampling: convergence
30 test runs, absolute and relative.

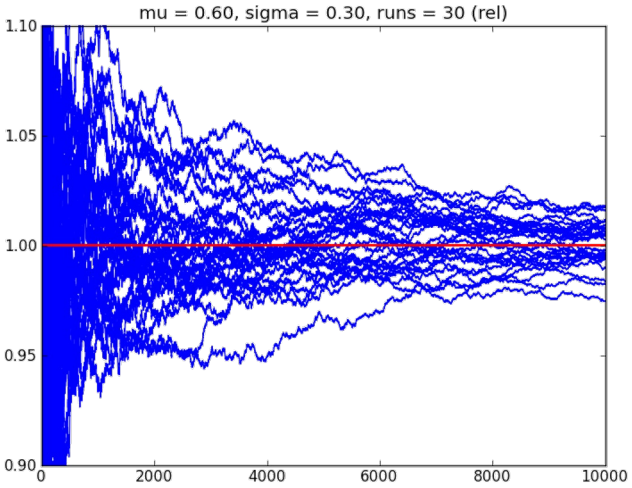












Importance sampling

- For the instrumental density *q(x)* one chooses a function which is easy to simulate
- "In principle", *q(x)* can be chosen arbitrarily, as long as the support is larger or equal to the support of *p(x)*:

$$\mathrm{supp}\,(q) \supseteq \mathrm{supp}\,(p)$$

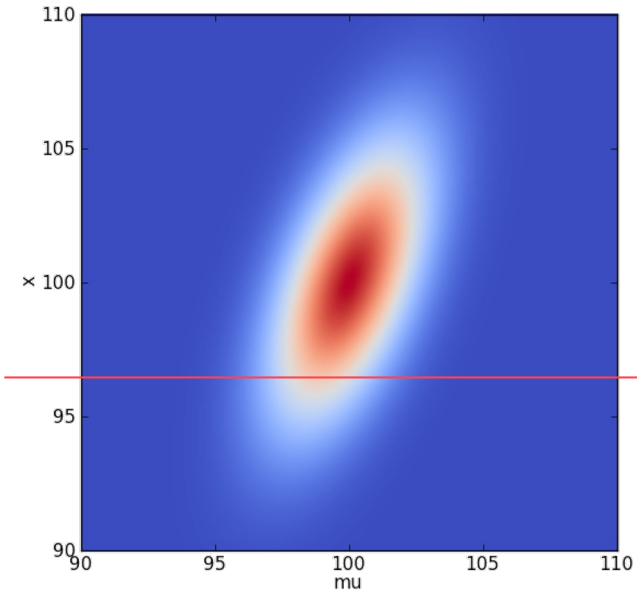
- However, the choice influences the convergence properties
- Note that the samples $x_j \sim q(x)$ can be re-used for different purposes (i.e., to compute different entities).

Application to our "yardstick" example

- Given:
  - Prior: $\mathrm{N}(\mu \mid \mu_0, \sigma_0^2)$
  - Likelihood: $\mathrm{N}(x \mid \mu, \sigma^2)$
- Sought for:
  - Posterior:

$$p(\mu \mid x) = \frac{\mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)}{\int \mathrm{N}(x \mid \mu', \sigma^2) \cdot \mathrm{N}(\mu' \mid \mu_0, \sigma_0^2) d\mu'}$$

x=97



Application to our "yardstick" example

- Problem: normalization required

$$p(\mu \mid x) = \frac{p(x \mid \mu)p(\mu)}{p(x)} = \frac{\mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)}{\int \mathrm{N}(x \mid \mu', \sigma^2) \cdot \mathrm{N}(\mu' \mid \mu_0, \sigma_0^2) d\mu'}$$

- It would be easier if the simulation can be carried out for non-normalized "densities" (target functions):

$$p(\mu \mid x) \propto \mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)$$

Importance sampling for non-normalized target functions

- Derivation is analogous to the case above. We want to compute the expectation:

$$\mathrm{E}[f] = \int f(x)p(x)dx$$

- Assume that the normalization factors are not known, neither for the instrumental density, nor for the target density:

$$p(x) = \frac{\tilde{p}(x)}{Z_p}$$

$$q(x) = \frac{\tilde{q}(x)}{Z_q}$$

$$\tilde{p}(x), \tilde{q}(x) \text{ not normalized}$$

Importance sampling for non-normalized target functions

- From this, it follows:

$$
\begin{aligned}
\mathrm{E}[f] &= \int f(x)\,p(x)\,dx \\
&= \int f(x)\,\frac{p(x)}{q(x)}\,q(x)\,dx \\
&= \frac{Z_q}{Z_p}\int f(x)\,\frac{\tilde{p}(x)}{\tilde{q}(x)}\,q(x)\,dx \\
&\approx \frac{Z_q}{Z_p}\cdot\frac{1}{m}\cdot\sum_{j=1}^{m} f(x_j)\,\frac{\tilde{p}(x_j)}{\tilde{q}(x_j)} \qquad \text{with } x_j \sim q(x) \\
&= \frac{Z_q}{Z_p}\cdot\frac{1}{m}\cdot\sum_{j=1}^{m} f(x_j)\cdot\tilde{w}_j
\end{aligned}
$$

- Problem: contains Z_q/Z_p , this quotient is not known

Importance sampling for non-normalized target functions

- Trick: use **the same samples** to compute this quotient
  - Evaluate the inverse of the quotient:

$$\begin{aligned}
\frac{Z_p}{Z_q} = \frac{1}{Z_q} \cdot Z_p &= \frac{1}{Z_q} \int \tilde{p}(x) dx \\
&= \frac{1}{Z_q} \int \frac{\tilde{p}(x)}{q(x)} q(x) dx \\
&= \int \frac{\tilde{p}(x)}{\tilde{q}(x)} q(x) dx \\
&\approx \frac{1}{m} \sum_{j=1}^{m} \frac{\tilde{p}(x_j)}{\tilde{q}(x_j)} = \frac{1}{m} \sum_{j=1}^{m} \tilde{w}_j \qquad \text{mit } x_j \sim q(x)
\end{aligned}$$

Importance sampling for non-normalized target functions

- Therefore, we obtain:

$$\mathrm{E}[f] \approx \sum_{j=1}^{m} f(x_j) \cdot w_j \quad \text{with} \quad w_j = \frac{\tilde{w}_j}{\sum_{k=1}^{m} \tilde{w}_k} = \frac{\tilde{p}(x_j)/\tilde{q}(x_j)}{\sum_{k=1}^{m} \tilde{p}(x_k)/\tilde{q}(x_k)}$$

- Equivalently:

$$\mathrm{E}[f] \approx \sum_{j=1}^{m} f(x_j) \cdot \tilde{w}_j \Big/ \sum_{k=1}^{m} \tilde{w}_k$$

- So the simple algorithm is:
  - Draw all samples $x_j \sim q(x)$
  - In each term inside the sum, weight by $\tilde{w}_j := \tilde{p}(x_j)/\tilde{q}(x_j)$
  - In the end, normalize using the sum of all weights: $\sum_{k=1}^{m} \tilde{w}_k$

Back to the "yardstick" example

- In this case: the (normalized) density is:

$$p(\mu \mid x) = \frac{\mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)}{\int \mathrm{N}(x \mid \mu', \sigma^2) \cdot \mathrm{N}(\mu' \mid \mu_0, \sigma_0^2) d\mu'}$$

- The non-normalized "density" (target function) is:

$$\tilde{p}(\mu \mid x) = \mathrm{N}(x \mid \mu, \sigma^2) \cdot \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)$$

- For example, the following instrumental densities can be used:
  - The prior density:

$$q(\mu) := \mathrm{N}(\mu \mid \mu_0, \sigma_0^2)$$

  - A normal density, centered on the simple mean of prior and measurement:

$$q(\mu) := \mathrm{N}(\mu \mid \frac{\mu_0 + x}{2}, \sigma_0^2)$$

Algorithm in the "yardstick" example

- Draw samples, e.g. from:

$$\mu_j \sim q(\mu) = \mathrm{N}(\mu \,|\, \frac{\mu_0 + x}{2}, \sigma_0^2)$$

- For each sample, compute the weight:

$$\tilde{p}(\mu_j) := \tilde{p}(\mu_j \,|\, x) = \mathrm{N}(x \,|\, \mu_j, \sigma^2) \cdot \mathrm{N}(\mu_j \,|\, \mu_0, \sigma_0^2)$$

$$\tilde{w}_j := \tilde{p}(\mu_j) / q(\mu_j)$$

(Note: *q(.)* is normalized in this case.)

- E.g. compute an approximation for the mean μ

$$\mathrm{E}[\mu] \approx \frac{\sum_{j=1}^{m} \mu_j \cdot \tilde{w}_j}{\sum_{j=1}^{m} \tilde{w}_j}$$

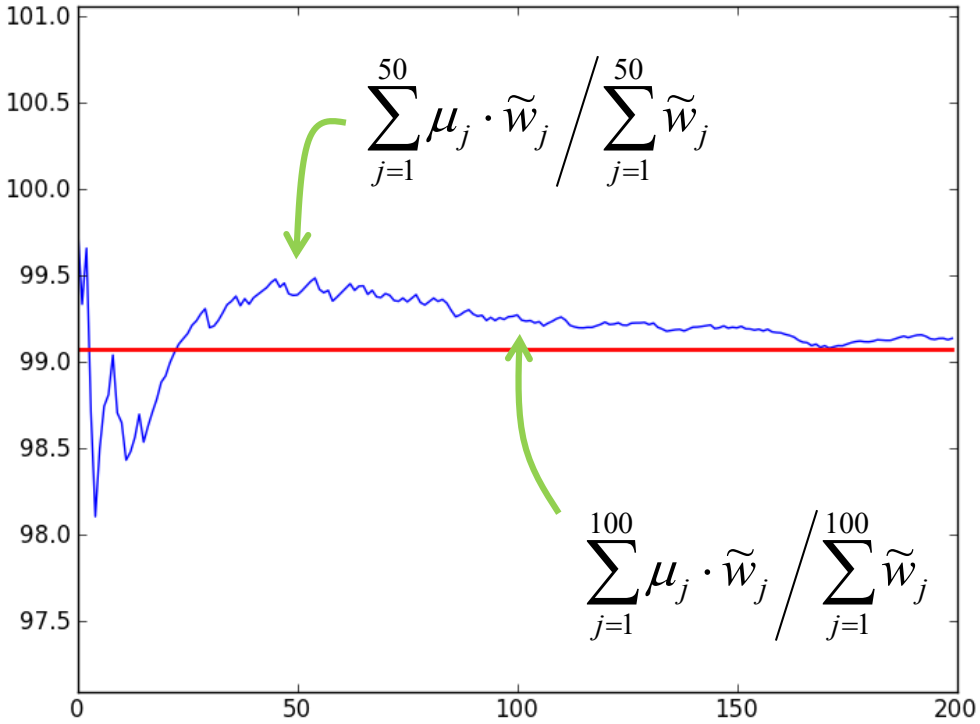
$$\mu_0 = 100$$
$$\sigma_0 = 2$$
$$x = 97$$
$$\sigma = 3$$

Simulation results (1)

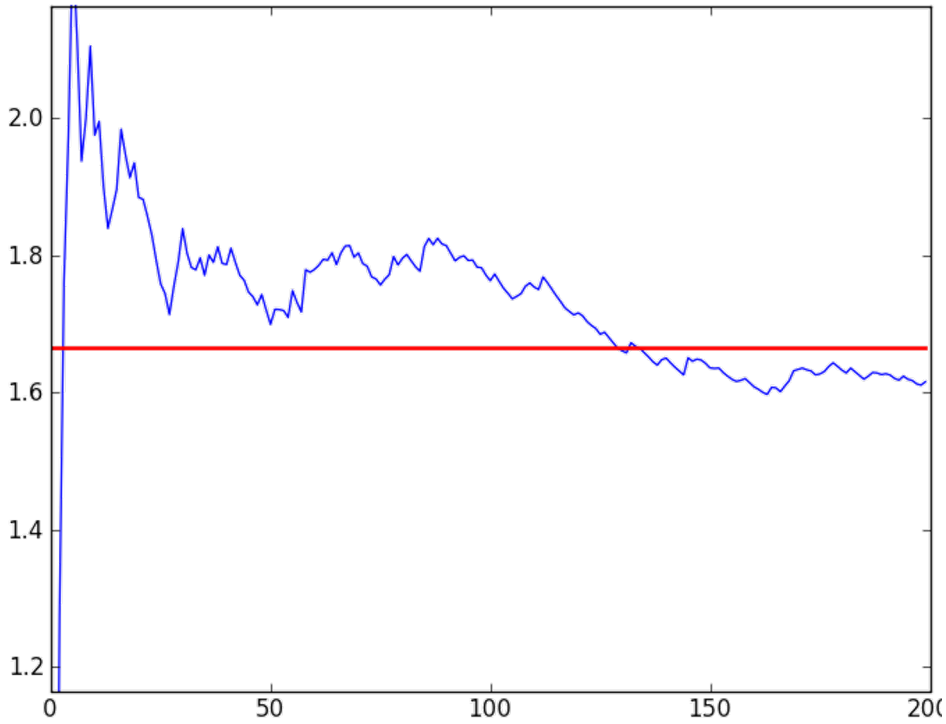
- Evolvement of the estimate, from m=1 to m=200 samples
- Correct values: $\mu_{\text{post}} = 99.0769$

  $\sigma_{\text{post}} = 1.66410$

Mean



Standard deviation

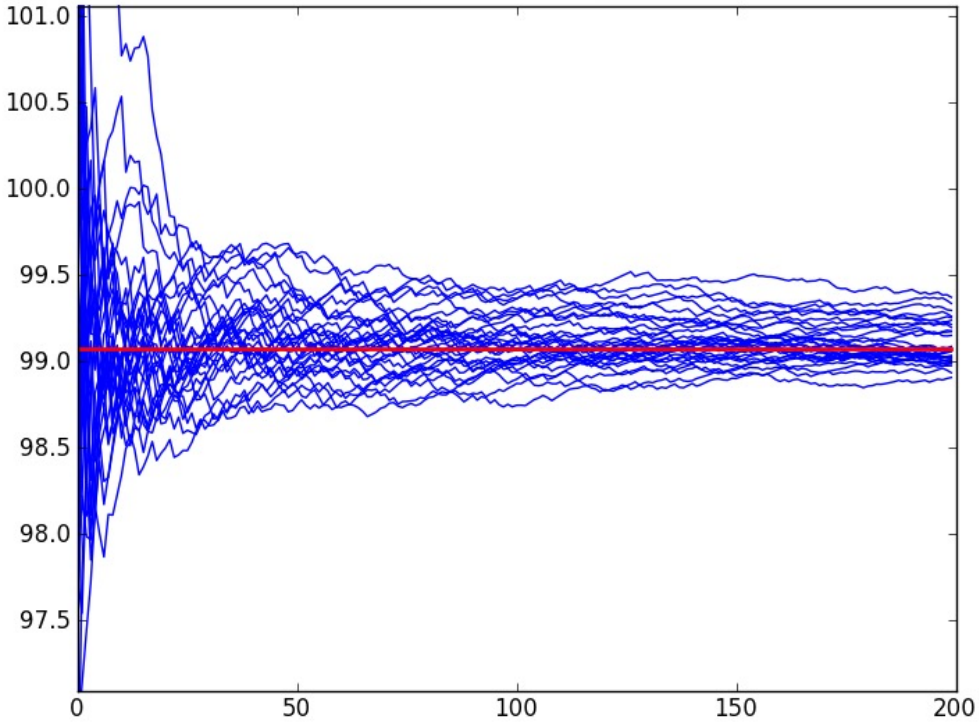


Simulation results (2)

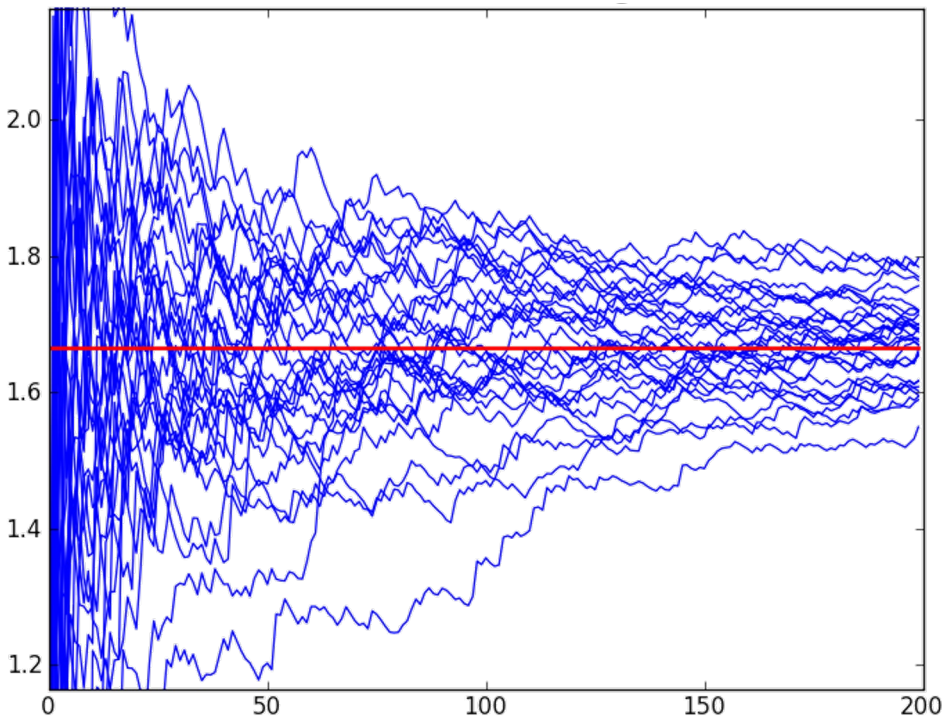
- 30 runs, each 200 samples
- Correct values: $\mu_{\text{post}} = 99.0769$

  $\sigma_{\text{post}} = 1.66410$

Mean



Standard deviation



Simulation results (3)

- Example: estimation of μ:
  - Nominal value: 99.076923
- Any single simulation can "arbitrarily" deviate from this
- However, the standard deviation of the simulation (see slide above: "How good is this approximation") is for m=200 samples:
  - 0.11766968
- In the special case of the 30 test runs above, we obtain empirically
  - mean (using all 30 test runs, each having 200 samples):
    99.096523 (relative deviation to nominal value: 0.0001978)
  - standard deviation:
    0.12475751

Further remarks for importance sampling

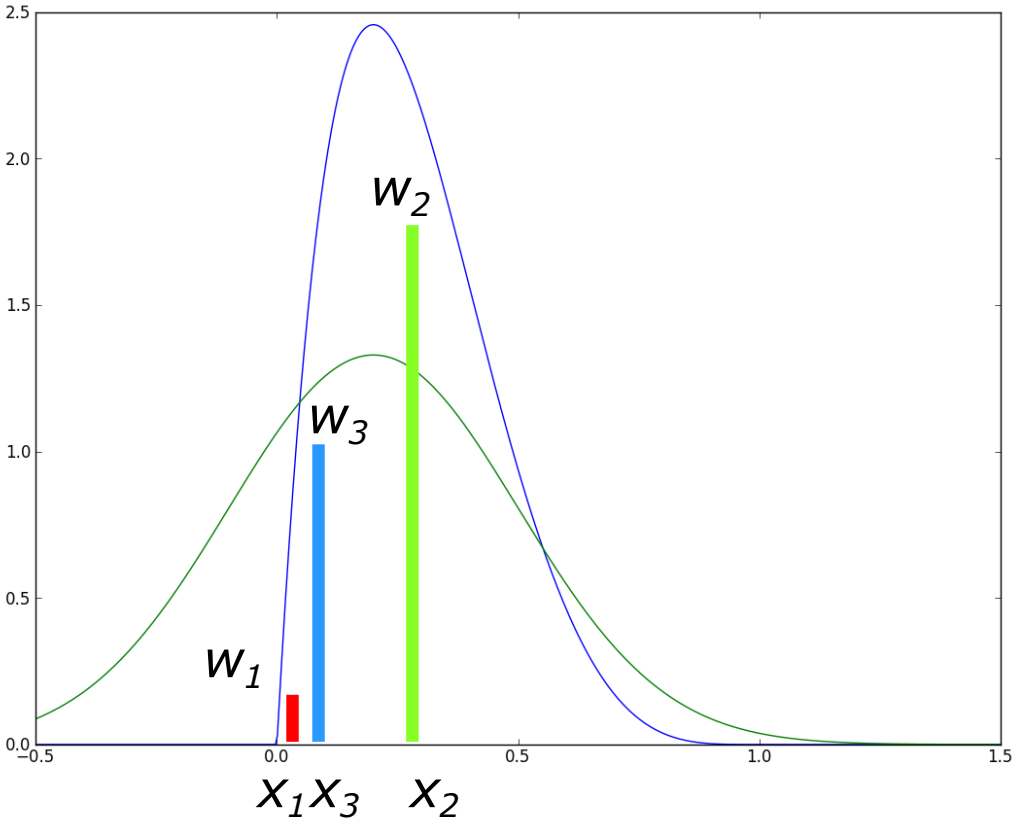
- Importance sampling does **not** generate samples $x_j \sim p(x)$!
- For this, one can use **SIR = „sampling importance resampling“**:
  - First, draw m samples from *q(x)*:

    $$x_j \sim q(x)$$
  - For each sample, compute the weight (as shown previously):

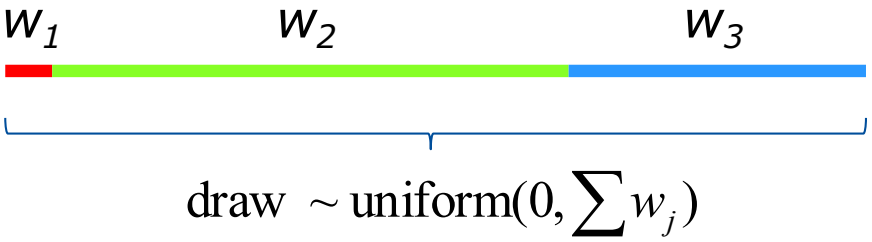
    $$w_j := p(x_j)/q(x_j)$$
  - Then, draw m new samples y_j from the x_j, with a probability $\sim w_j$
  - The samples y_j obtained in this manner are (approximately) distributed according to *p(x)*

    $$y_j \sim p(x)$$
  - (Note: normally, these samples will contain duplicates, since they are obtained by “sampling with replacement”.)

SIR



Resampling: $p(x_i) = \frac{w_i}{\sum_j w_j}$



Example result: {x_2, x_3, x_2}

SIR is part of the "particle filter" algorithm in robotics!

What we have learned so far

- We want to compute expectation values and/or maxima of distributions *p(x)*

$$\mathrm{E}[f] = \int f(x)\,p(x)\,dx \qquad \text{or} \qquad x^* = \arg\max_x p(x)$$

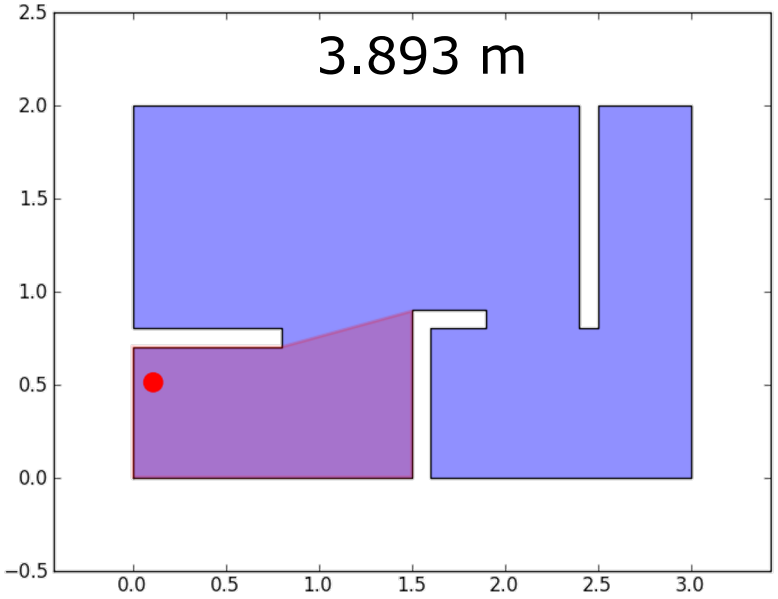
- For simple distributions, there are existing methods
  - Uniform distribution, inversion trick
- In other cases: approximation using a sample based approach:
  - Accept/ reject (hard: need to find constant m)
  - Importance sampling (only useful for expectations, not for finding the maximum)
- The algorithms also work if the target functions are not normalized (i.e. are no distributions)

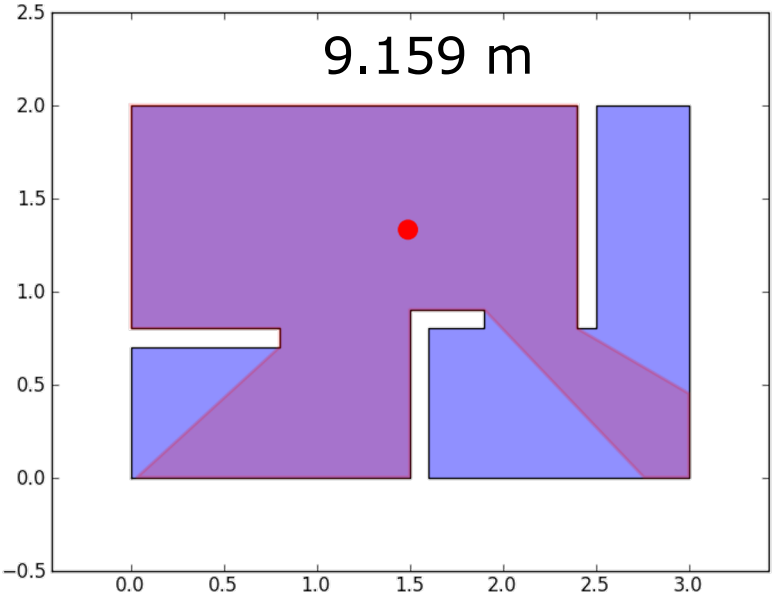
Example: Art Gallery Problem

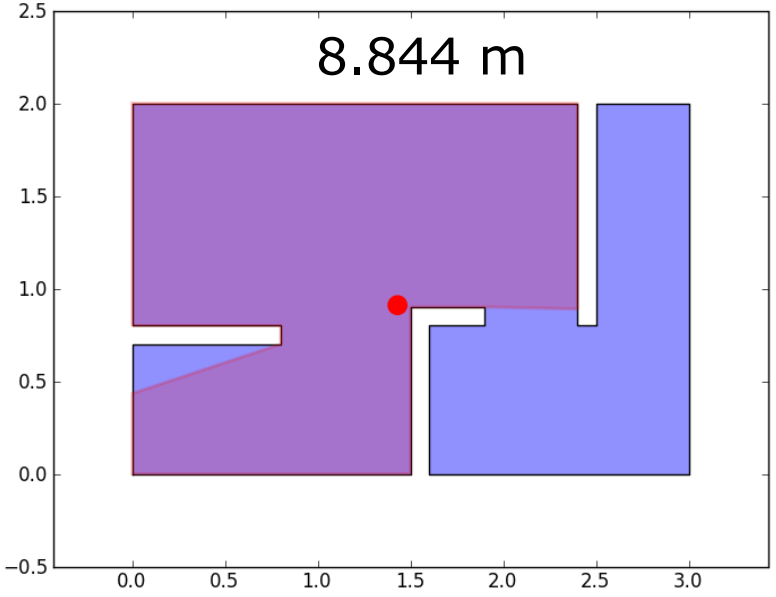
Example: art gallery problem

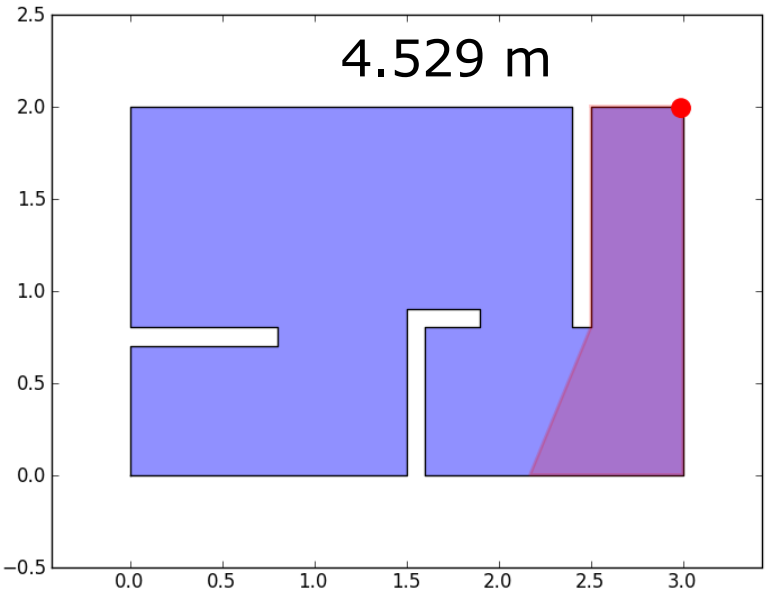
- This example is similar to the well-known “art gallery” problem
- Here:
  - A 360° camera shall be installed in an art gallery
  - The goal is to choose the position in such a way that the monitored area is largest
  - Main criterion: “total length of monitored wall”
- The target function is not a distribution (not normalized)
- In this case: **search for the (global) maximum** (not: expectation)

Art gallery problem: total wall length 16.4 m

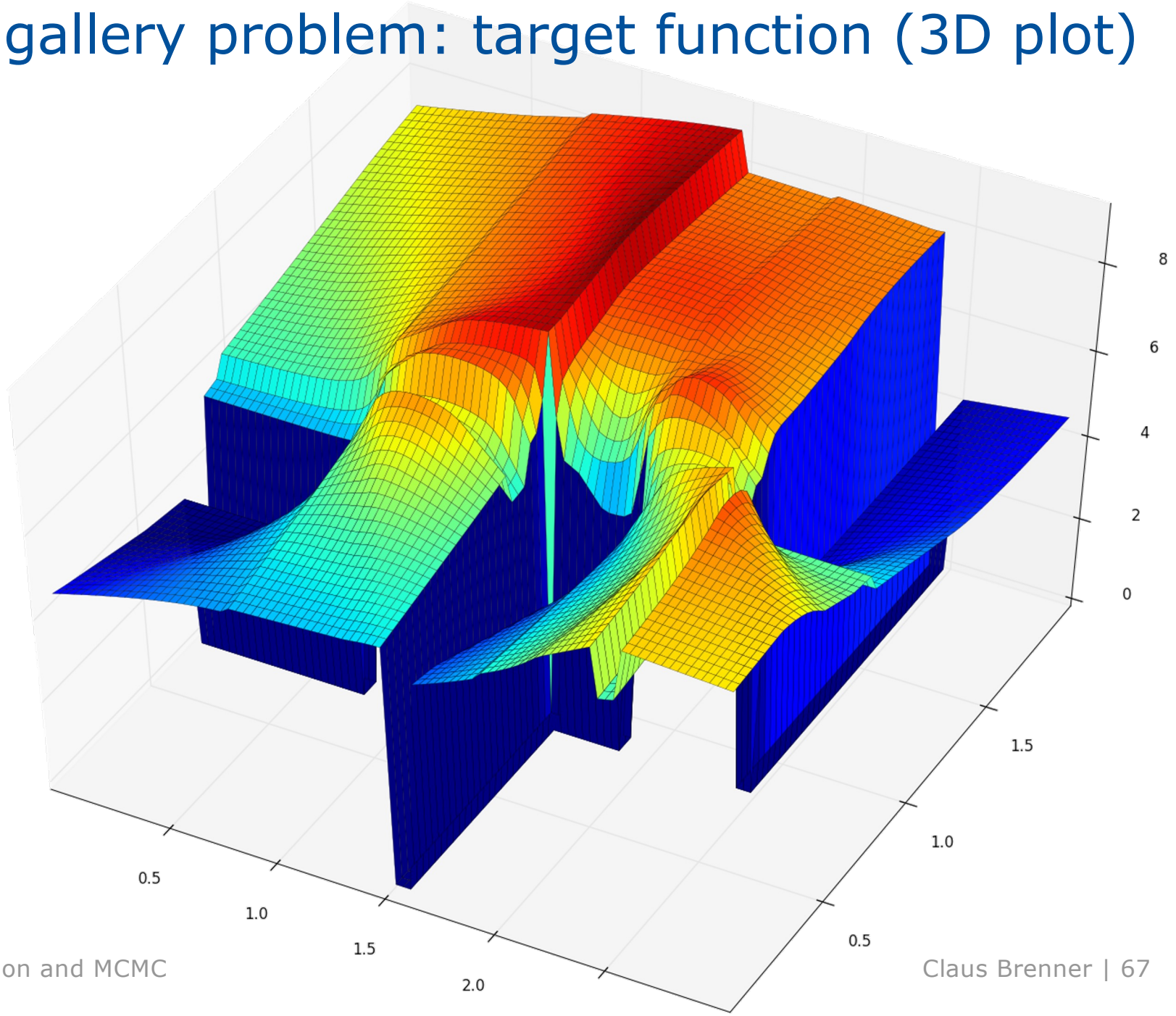




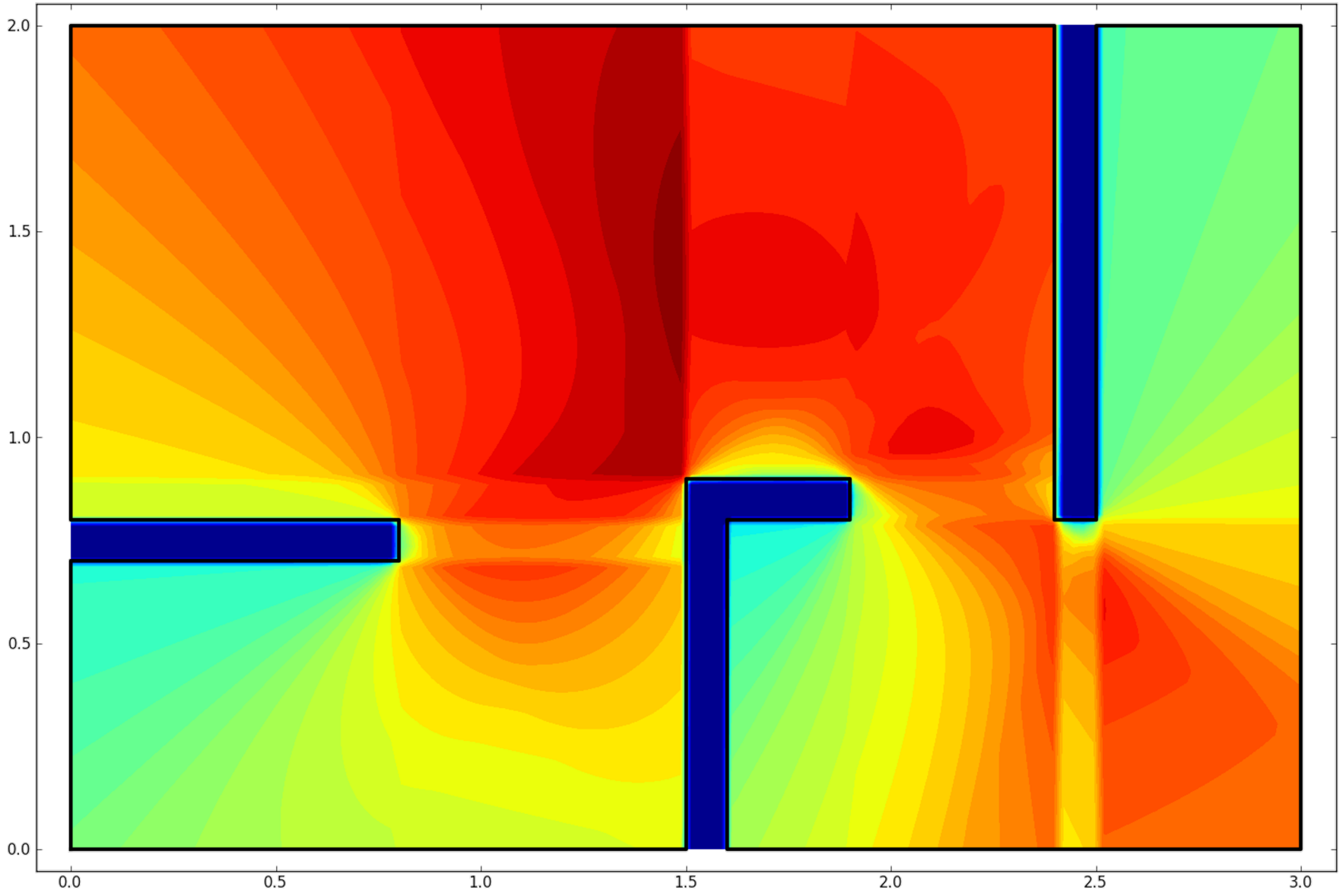




Art gallery problem: target function (3D plot)



Art gallery problem: target function (2D plot)

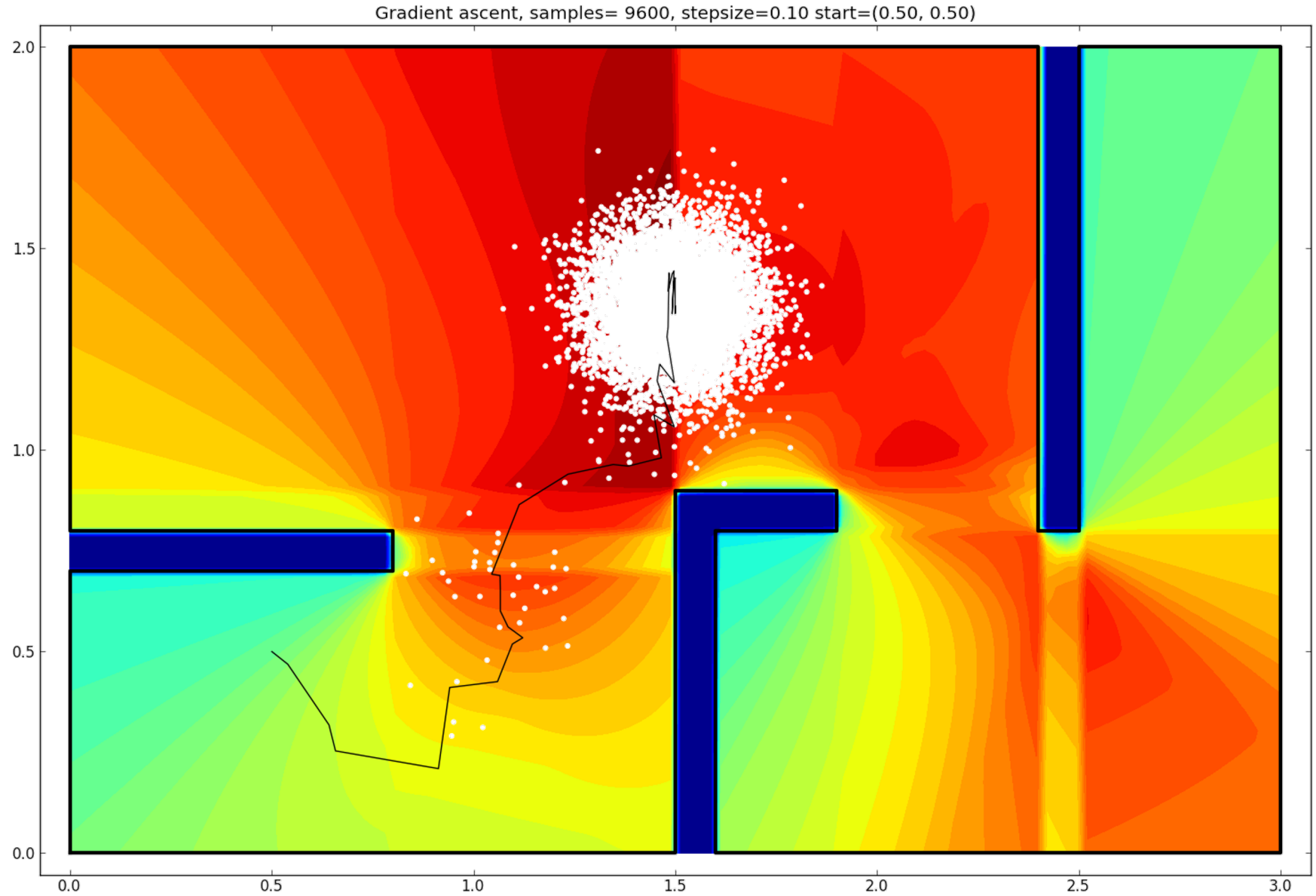


Art gallery problem: finding the maximum

- The target function is not given in analytical form
- However, using a "visibility" algorithm, the function value can be evaluated for any given camera location

- First idea: gradient ascent method
  - Start with an arbitrary camera location
  - Evaluate the target function
  - Since the gradient is not known analytically →
    - Propose a random step (random length and direction)
    - Evaluate the target function there as well
    - Accept if the value increases.

Art gallery: gradient ascent, favorable initial value



(white: proposals which were not accepted)

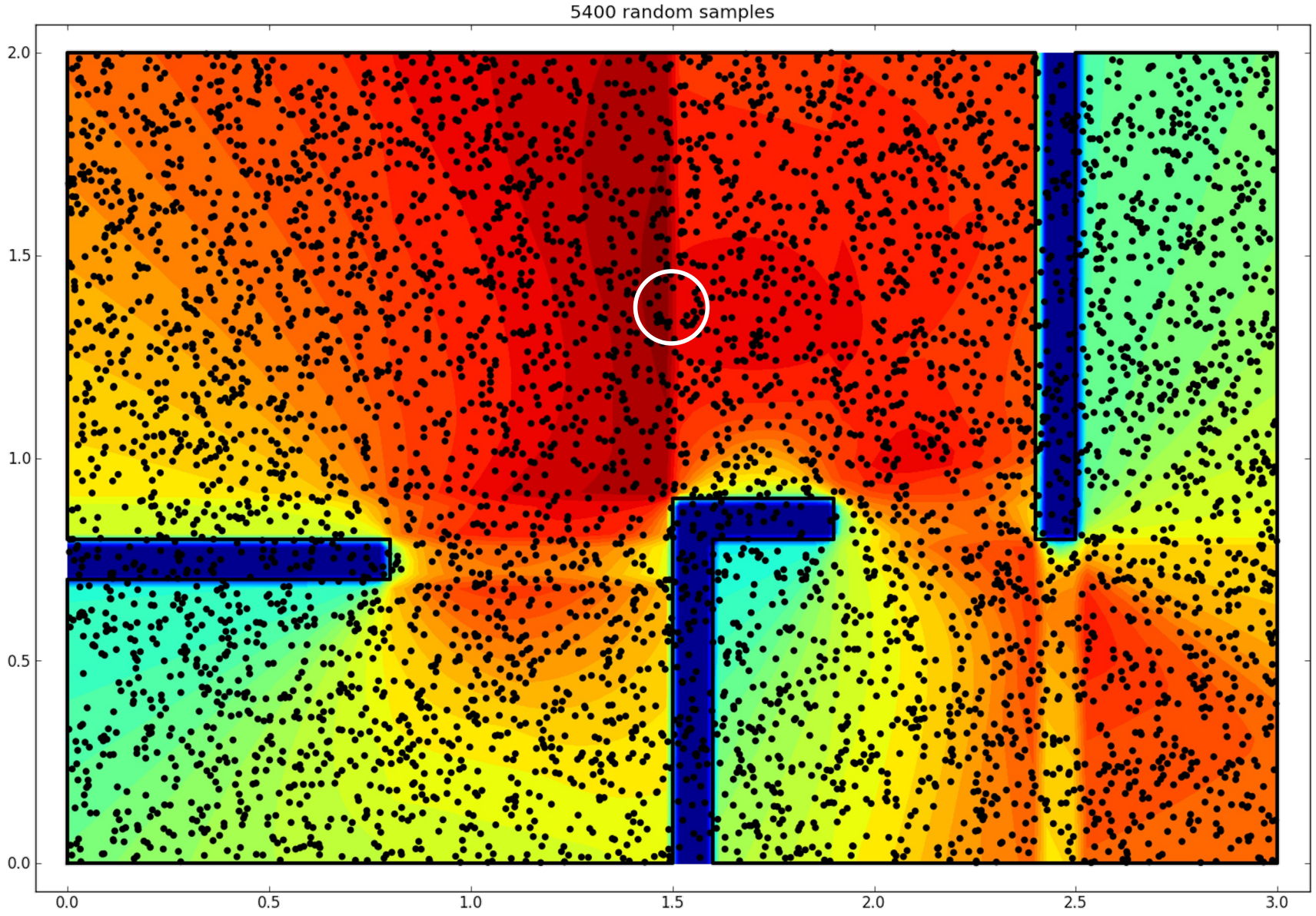
Art gallery: gradient ascent, unfavorable initial value

(white: proposals which were not accepted)

Art gallery problem, another try: uniform sampling

- Finding from previous experiment:
  - Gradient ascent may get caught in local maximum
- Another try: exhaustive simulation
  - In the present case we have the advantage that our search space is bounded (the possible camera locations must be inside the gallery): [0, 3] x [0, 2]
  - Draw uniform samples (x, y) in this area
  - Evaluate target function for each sample
  - Compute the maximum of all samples.

Art gallery: exhaustive, uniform samples



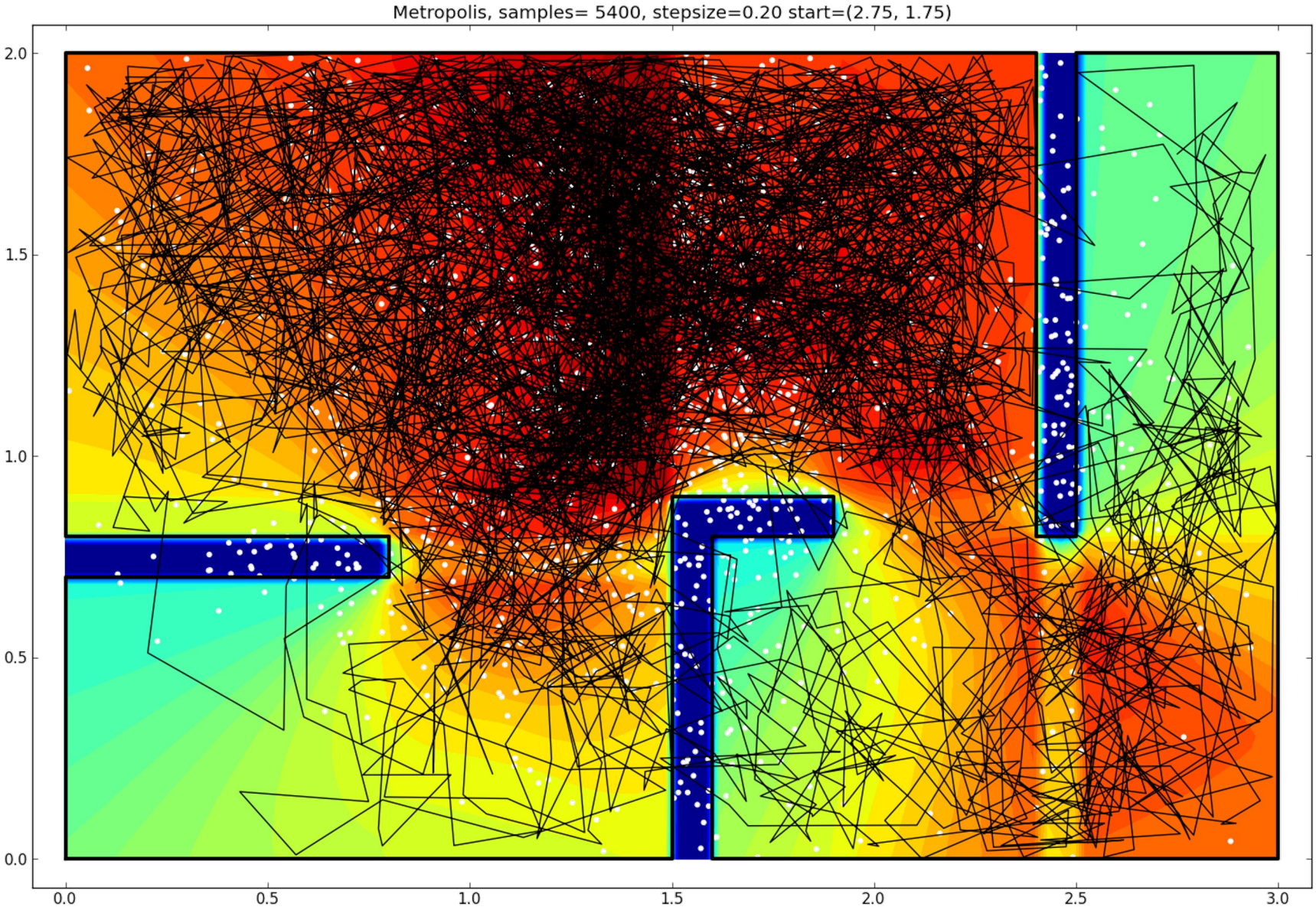
Art gallery: disadvantages of uniform sampling

- Uniform sampling works
  - Precondition: the parameter space must be bounded
- **But**: the density of the samples is not adapted to our target function
- Therefore, in areas where the target function is small, many samples are wasted
- Remember each single sample requires the evaluation of the target function (in our case, wall visibility) → costly.

Art gallery: MCMC

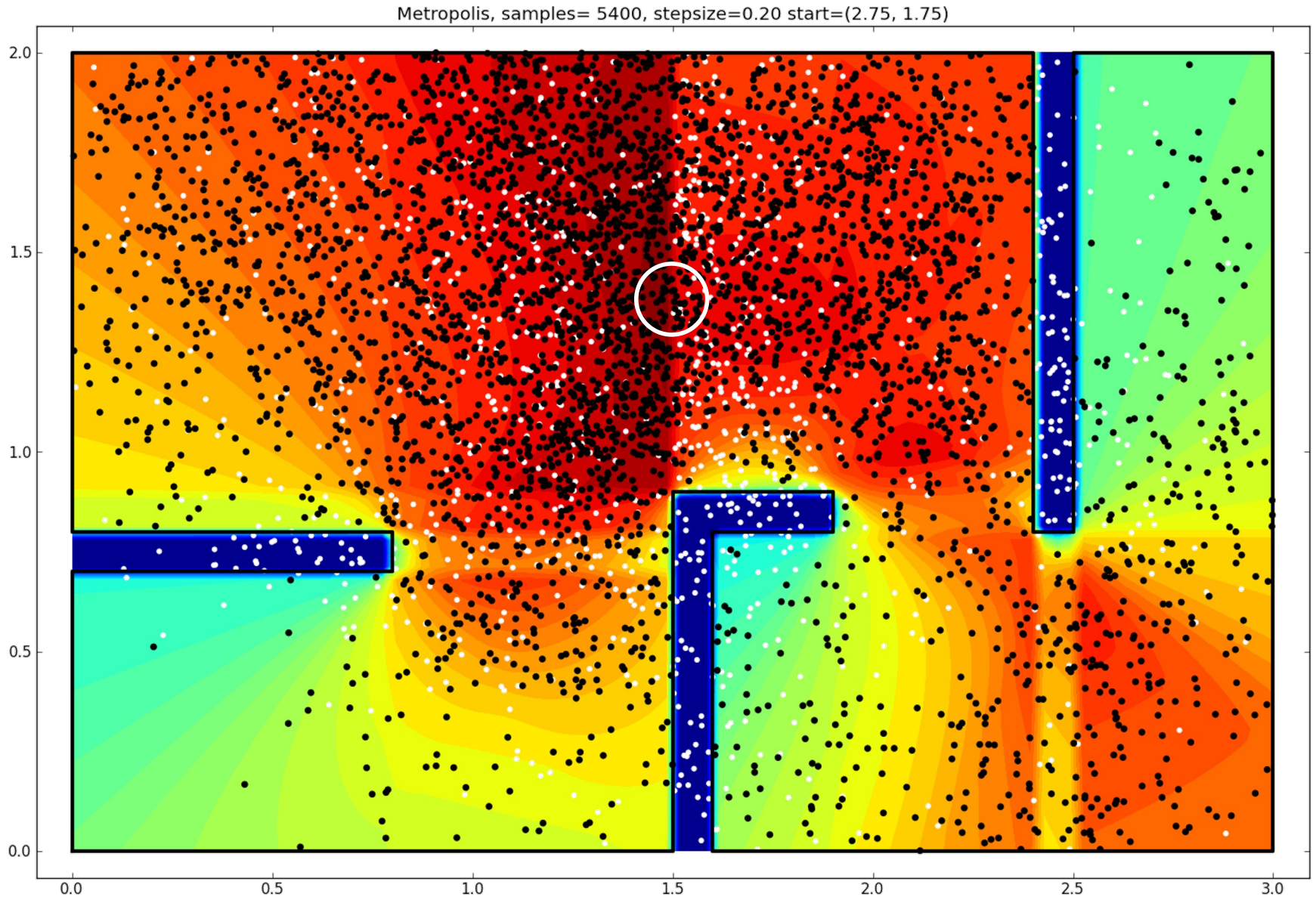
- Idea:
  - If it were possible to draw the samples from the target distribution rather than uniformly, then
    - less samples would be wasted in areas where the target function is small
    - in turn, more samples would be available in areas where the target function is large
  - This would increase the probability to obtain a value close to the maximum
- One possible solution is **Markov chain Monte Carlo (MCMC)** sampling
- (Details in the next lecture.)

Art gallery: MCMC



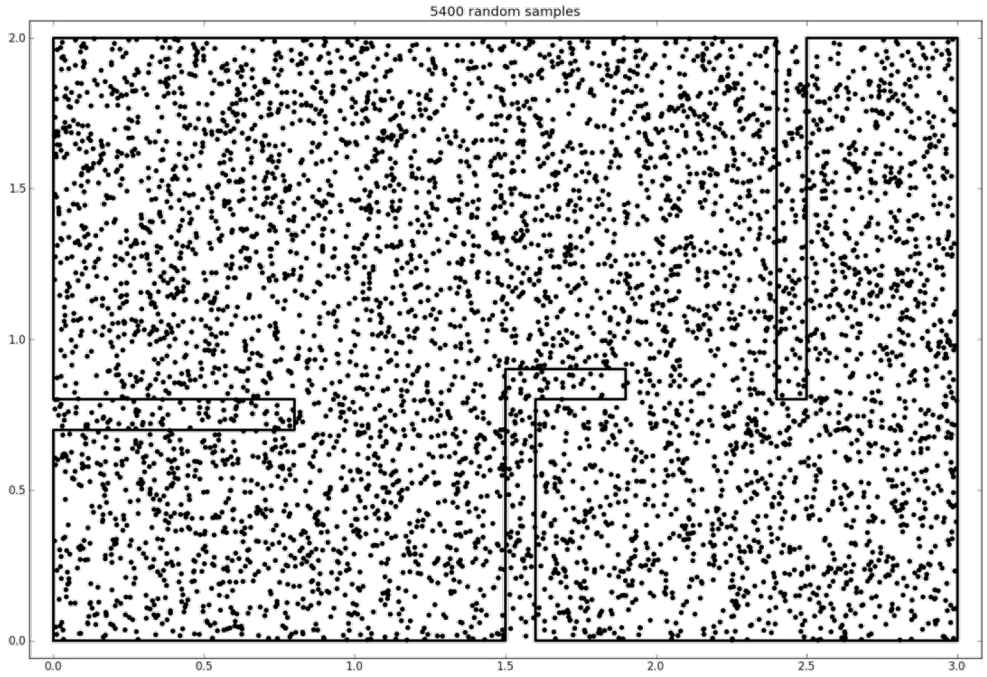
(white: proposals which were not accepted)

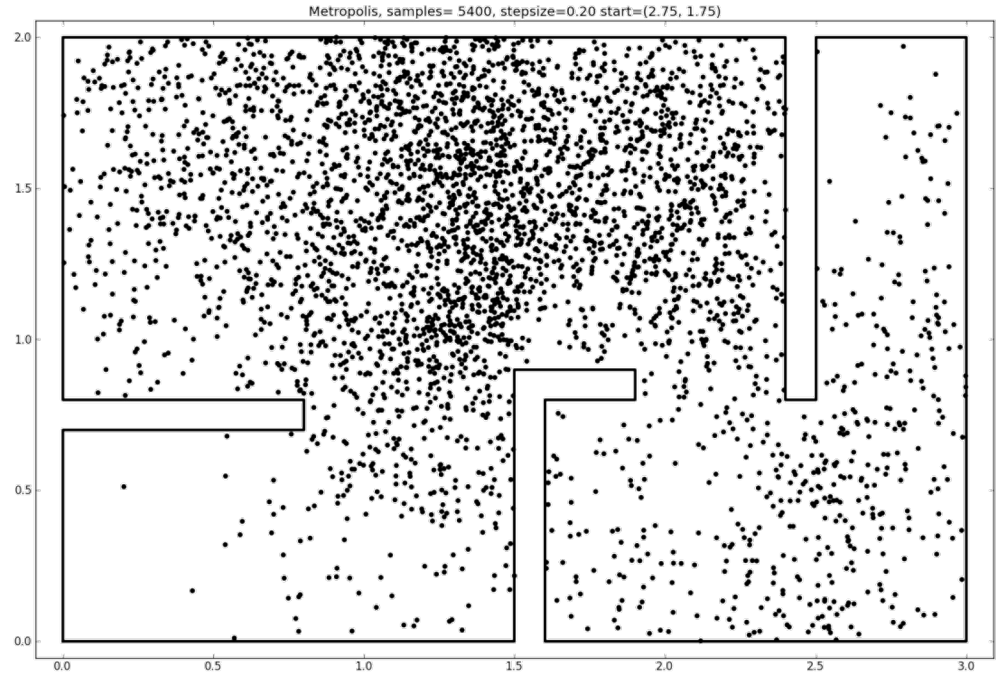
Art gallery: MCMC

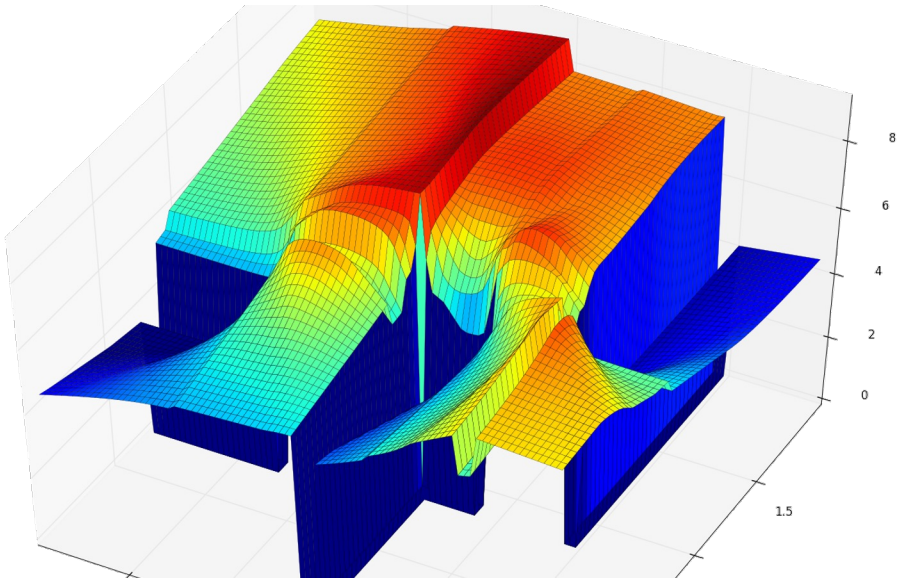


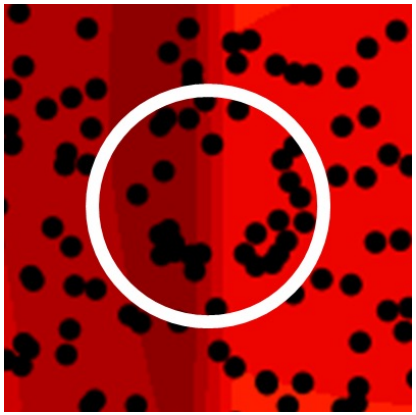
(white: proposals which were not accepted)

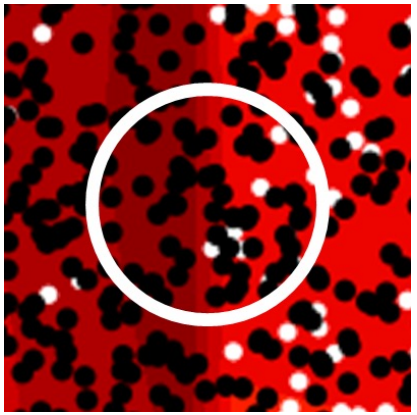
Art gallery: comparison uniform ←→ MCMC











Comparison of all methods

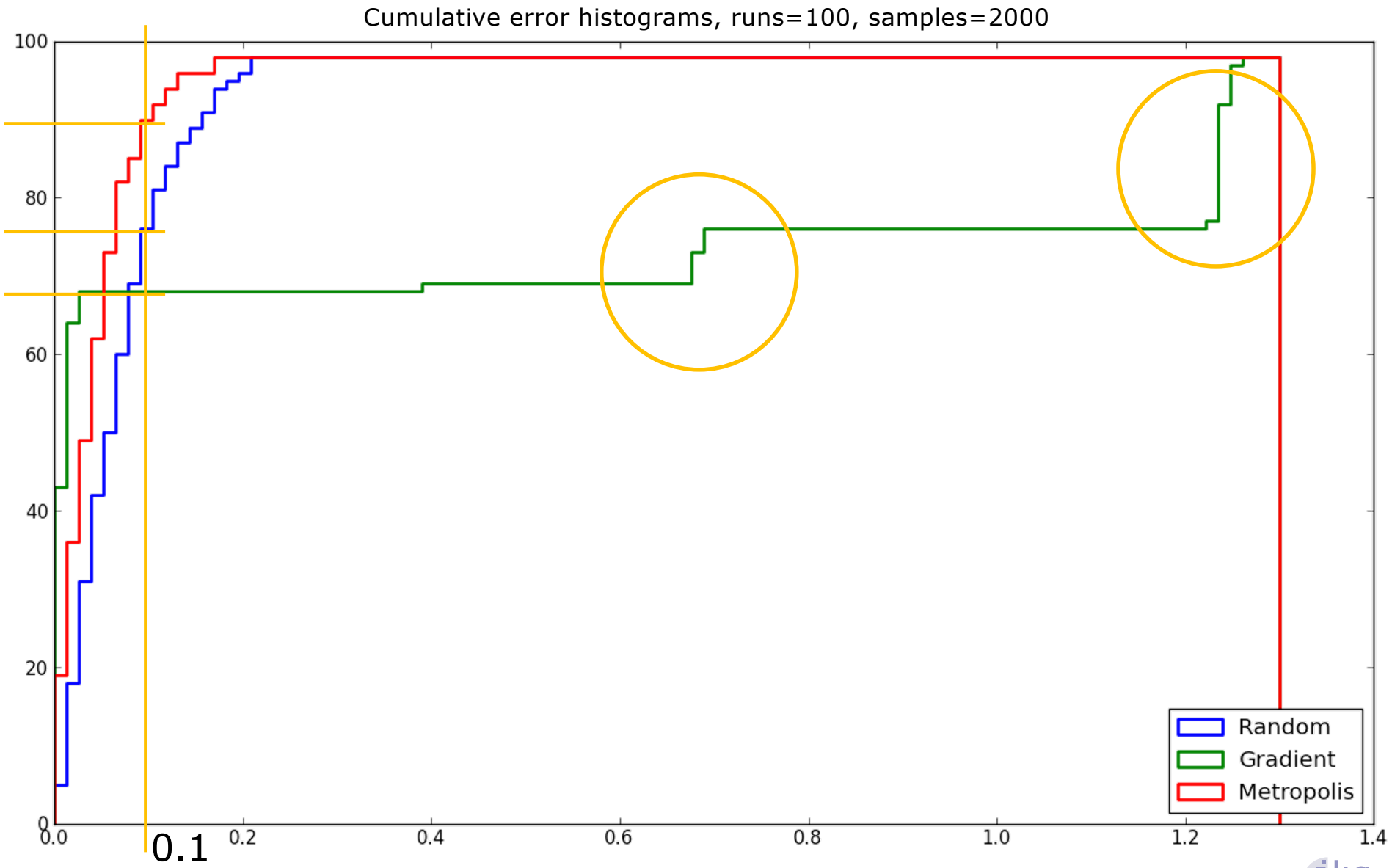
- Which method is better?
  - All are randomized methods
  - → Even if MCMC has a higher density of samples in the vicinity of the global maximum than "uniform sampling", it can well be that the "uniform" method has a better "random hit"
  - Therefore, single experiments can't be compared
  - Example test run:

```
Grid evaluation using 90 x 60 = 5400 points
  Max at 1.48314607865 1.35593184746 value 9.14237326216
Random sampling: 5400 samples
  Max at [ 1.49902129  1.33596585] value 9.18470874715
Gradient ascent: 5400 samples, start_pos [ 2.75  1.75] stepsize 0.1
  Max at [ 2.50022098  0.59974736] value 8.34898277747
  Number of moves: 29
Metropolis: 5400 samples, start_pos [ 2.75  1.75] stepsize 0.2
  Max at [ 1.4990767   1.33295116] value 9.18477171409
  Number of moves: 3825
```

Comparison of all methods

- Test: 100 experiments
  - A: draw a random initial point and iterate 2000 steps of “gradient ascent”, take the last point
  - B: draw 2000 samples uniformly distributed, determine the maximum
  - C: draw random initial point, use MCMC sampling for 2000 steps, determine the maximum
- Assess the results using cumulative histograms of the distance to the correct solution.

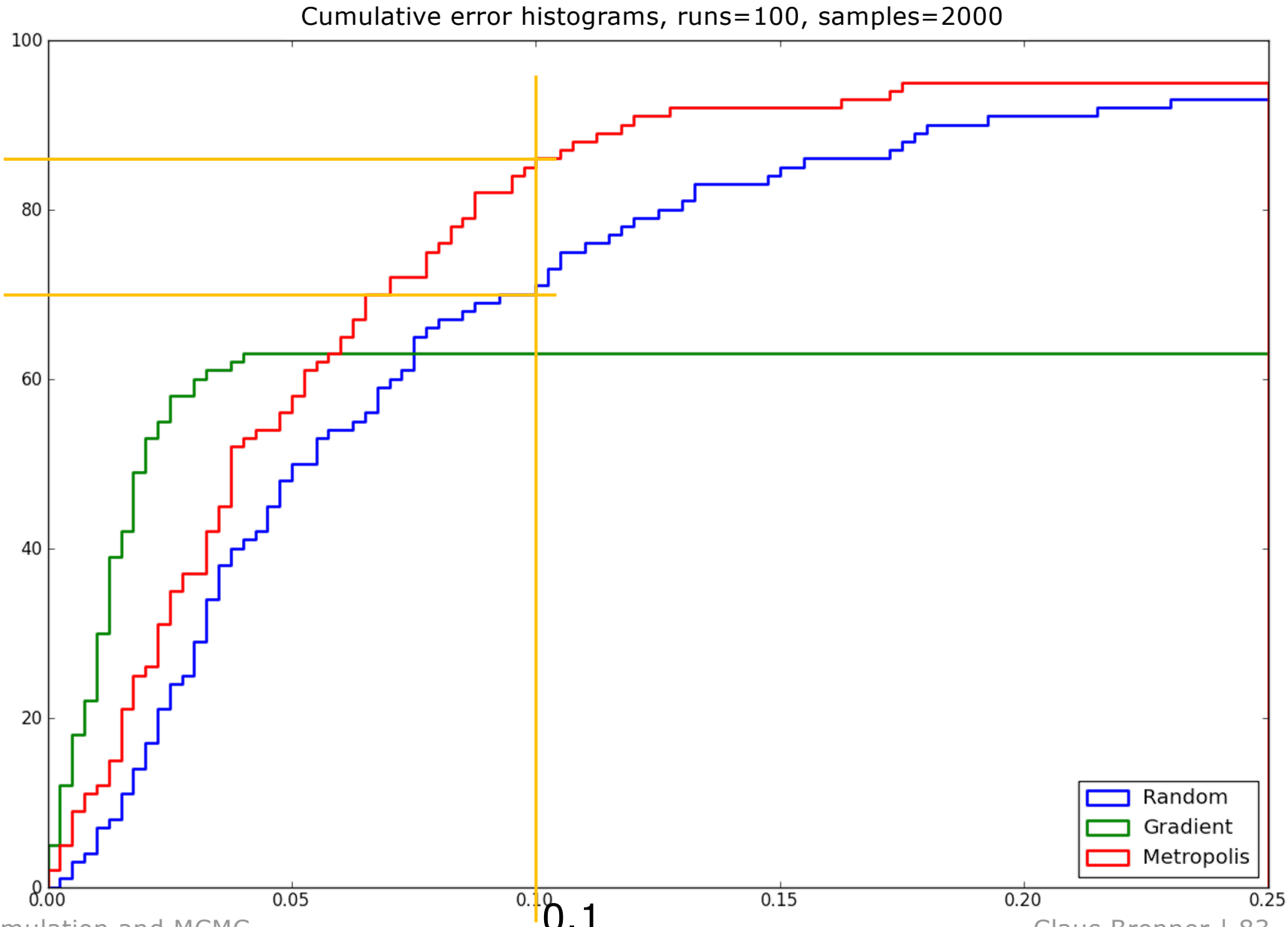
Gradient ascent vs. uniform vs. MCMC



Explanation for the diagram (and yellow lines)

- How many of the 100 simulations reach a solution which is closer than 0.1 (distance in the xy plane) from the global maximum?
  - Uniformly distributed: ~76
  - Gradient ascent: ~68
    - Gradient ascent outperforms both other methods for very small distances
    - For some initial positions (approximately 30%) only solutions in local maxima are found (at distances 0.7 and 1.25)
  - **MCMC: ~90**

Zoom view for distances 0 .. 0.25



0.1

Conclusion for the “art gallery” problem

- Single simulation results are not meaningful, since these are randomized methods
- However, over many test runs, one can conclude:
  - MCMC generates more solutions close to the optimum than uniform sampling
  - Uniform sampling can only be used if the parameter space is bounded
  - Gradient ascent can be very good, but only if a favorable initial value is chosen. For complex functions with many local maxima, the method is not applicable.

References

- C. M. Bishop: Pattern Recognition and Machine Learning, Springer 2006 (Chap. 11)
- C. P. Robert, G. Casella: Monte Carlo Statistical Methods, Springer 2004
- W. R. Gilks, S. Richardson, D. J. Spiegelhalter: Markov Chain Monte Carlo in Practice, Chapman & Hall/CRC 1996