Advanced computational statistics, lecture 7

Frank Miller, Department of Computer and Information Science, Linköping University

Department of Statistics; Stockholm University

May 17, 2023

Course schedule

- Topic 1: **Gradient based optimisation**
- Topic 2: **Stochastic gradient based optimisation**
- Topic 3: **Gradient free optimisation**
- Topic 4: **Optimisation with constraints**
- Topic 5: **EM algorithm and bootstrap**
- Topic 6: **Simulation of random variables**
- Topic 7: **Numerical and Monte Carlo integration; importance sampling**

Course homepage: http://www.adoptdesign.de/frankmillereu/adcompstat2023.html

Includes schedule, reading material, lecture notes, assignments

Today’s schedule

- Numerical integration
  - Newton-Côtes rules
  - Gaussian quadrature
- Importance sampling
- Antithetic sampling
  - Combining importance and antithetic sampling

Integration in Statistics

- Expected value: $E(X) = \int_{-\infty}^{\infty} x \cdot f(x)dx$
- Variance: $\mathrm{Var}(X) = \int_{-\infty}^{\infty}(x - E(X))^2 \cdot f(x)dx$
- Probabilities for distributions with given density:

$$P(X \leq y) = \int_{-\infty}^{y} f(x)dx$$

- The likelihood function might be an integral, e.g. in mixed effect models like in the Alzheimer's example by Givens and Hoeting, ch.5:

$$L\left(\beta, \sigma_\gamma^2 \middle| y\right) = \prod_{i=1}^{22} \int \left[\phi(\gamma_i; 0, \sigma_\gamma^2) \prod_{j=1}^{5} f(y_{ij}|\lambda_{ij})\right] d\gamma_i$$

where ϕ is normal density and f Poisson density

Integration in Statistics

- Analytical integration (in rare cases…)
- Numerical integration (Evaluation of integrant at a finite number of points and compute weighted sum)
- Using Monte Carlo methods

One-dimensional numerical integration

- Computation of $\int_a^b f(x)dx$

- Divide first [a,b] into n subintervals $[x_i, x_{i+1}]$, i=0,…,n-1 ($a=x_0$, $b=x_n$); then $\int_a^b f(x)dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x)dx$

- Use a "simple rule" by choosing m+1 nodes x_{ij}^* in $[x_i, x_{i+1}]$ and approximate $\int_{x_i}^{x_{i+1}} f(x)dx \approx \sum_{j=0}^{m} A_{ij} f\left(x_{ij}^*\right)$

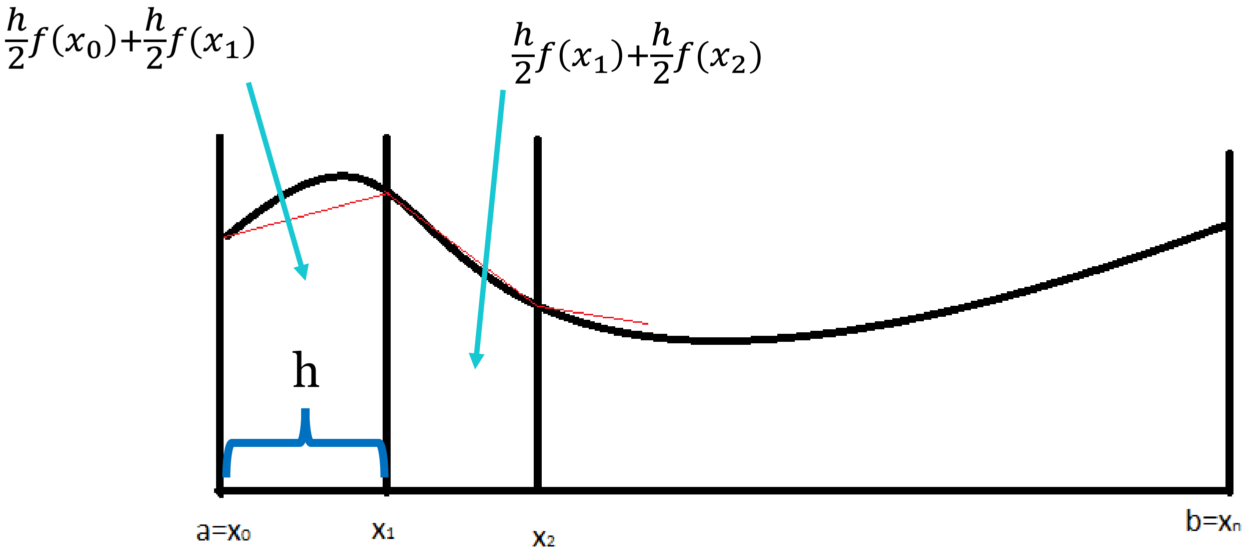
Newton-Côtes rules

- Computation of $\int_{x_i}^{x_{i+1}} f(x)dx$ by $\sum_{j=0}^{m} A_{ij} f\left(x_{ij}^*\right)$

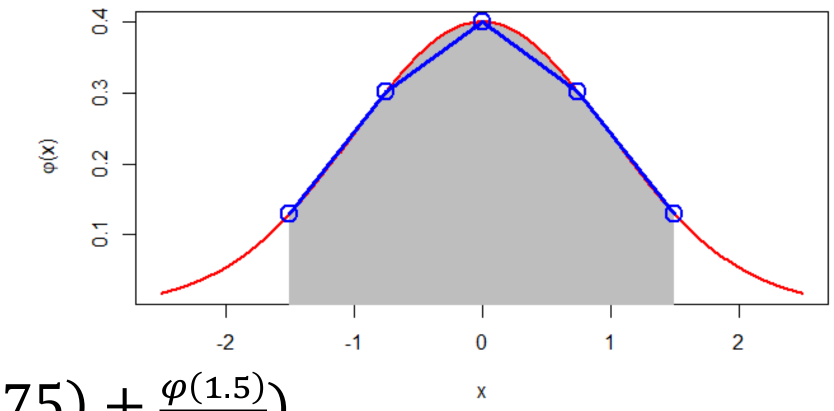
- m+1 equally spaced nodes x_{ij}^* in $[x_i, x_{i+1}]$
- Riemann rule (m=0): $x_{i0}^* = x_i,\, A_{i0} = (x_{i+1} - x_i)$
- Trapezoidal rule (m=1): $x_{i0}^* = x_i, x_{i1}^* = x_{i+1},\, A_{i0} = A_{i1} = \frac{x_{i+1}-x_i}{2}$
- Simpson's rule (m=2): $x_{i0}^* = x_i, x_{i1}^* = \frac{x_i+x_{i+1}}{2}, x_{i2}^* = x_{i+1},$
  $A_{i0} = A_{i2} = \frac{x_{i+1}-x_i}{6}, A_{i1} = 4 \cdot \frac{x_{i+1}-x_i}{6}$
- Compare Givens and Hoeting, Figure 5.2

Newton-Côtes rules: Trapezoidal rule

- Computation of $\int_a^b f(x)dx$
- We use equally spaced x_i, i.e. $x_i = ih + a, h = \frac{b-a}{n}$
- Then the trapezoidal rule becomes: $\int_a^b f(x)dx \approx \frac{h}{2}f(a)+h\sum_{i=1}^{n-1} f(x_i)+\frac{h}{2}f(b)$



Trapezoidal rule: Example



- X standard normal distributed
- Compute P(-1.5<X<1.5)= $\int_{-1.5}^{1.5}\varphi(x)dx$ with $\varphi(x)=\frac{1}{\sqrt{2\pi}}e^{-x^2/2}$ using the trapezoidal method
- n=4: $\int_{-1.5}^{1.5}\varphi(x)dx \approx \frac{3}{4}(\frac{\varphi(-1.5)}{2}+\varphi(-0.75)+\varphi(0)+\varphi(0.75)+\frac{\varphi(1.5)}{2})$ $=\frac{3}{4}(0.1295/2+0.3011+0.3989+0.3011+0.1295/2)=0.8481$
- Iterative application of the trapezoidal rule:
- To obtain in a next step a better approximation, use n=8, compute additionally $\varphi(-1.125), \varphi(-0.375), \varphi(0.375), \varphi(1.125)$, and $\frac{3}{8}(\frac{\varphi(-1.5)}{2}+\varphi(-1.125)+\varphi(-0.75)+\cdots+\varphi(1.125)+\frac{\varphi(1.5)}{2})$
- Do this until stopping criterion met
- A relative stopping criterion is reasonable here

Trapezoidal rule: Example

- With a relative stopping criterion $cc = \left|\frac{Integral}{Integral-old} - 1\right| < 10^{-6}$, we obtain following approximations of the integral:

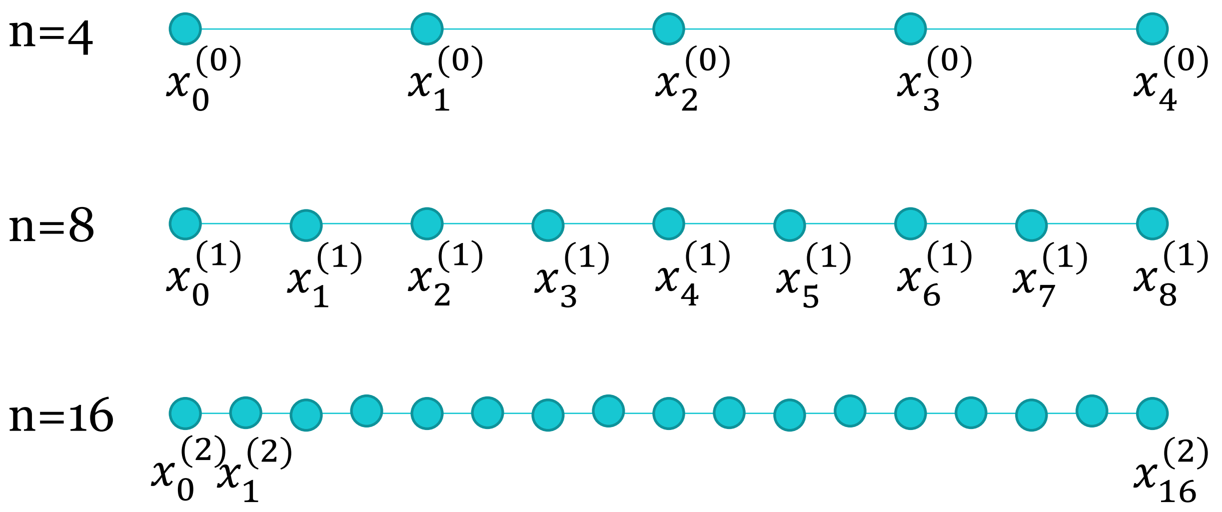
nodes	integr-ap	log_10(cc)
4	0.8480511	
8	0.8618243	-1.7893847
16	0.8652468	-2.4010844
32	0.8661010	-3.0055700
64	0.8663144	-3.6082363
128	0.8663678	-4.2104480
256	0.8663812	-4.8125460
512	0.8663845	-5.4146154
1024	0.8663853	-6.0166778

This means that $cc = 10^{-1.789}$

- Using pnorm-function:

```
pnorm(1.5) - pnorm(-1.5) = 0.8663856
```

Iterative application of trapezoidal rule



- Faster if one reuses already computed values for next iteration

Gaussian quadrature

- Newton-Côtes rules based on equidistant nodes
- Gaussian quadrature uses idea that it might be better to be more flexible and allow arbitrary distances between nodes x_i and corresponding weights A_i to compute

$$\int_a^b f(x)dx \approx \sum_{i=0}^{m} A_i f(x_i)$$

- Gaussian quadrature is defined for given weight function w(x)

$$\int_a^b f(x)w(x)dx \approx \sum_{i=0}^{m} A_i f(x_i)$$

- For $w(x) = e^{-x^2}$: "Gauss-Hermite" (note: Givens and Hoeting use Gauss-Hermite with $w(x) = e^{-x^2/2}$)

Gauss-Hermite quadrature

- Gauss-Hermite quadrature uses $w(x) = e^{-x^2}$ and can integrate from -∞ to +∞.
- E.g. for m+1=7 nodes, x_i and A_i are in following table:

x_i	-2.652	-1.674	-0.816	0	0.816	1.674	2.652
A_i	0.001	0.055	0.426	0.810	0.426	0.055	0.001

- Given a function f(x) and f^*(x)=f(x)/w(x), we approximate the integral by

$$\int_{-\infty}^{\infty} f(x)dx = \int_{-\infty}^{\infty} f^*(x)w(x)dx \approx \sum_{i=0}^{6} A_i f^*(x_i)$$

Gauss-Hermite quadrature – Example

x_i	-2.652	-1.674	-0.816	0	0.816	1.674	2.652
A_i	0.001	0.055	0.426	0.810	0.426	0.055	0.001

- $f^*(x)=f(x)/w(x)$, $\int_{-\infty}^{\infty} f(x)dx = \int_{-\infty}^{\infty} f^*(x)w(x)dx \approx \sum_{i=0}^{6} A_i f^*(x_i)$ with $w(x) = e^{-x^2}$
- Example: $f(x) = \frac{1}{\sqrt{\pi}} e^{-x^2}$: Compute numerically integral from -∞ to +∞ with Gauss-Hermite and m=6 (we know that this should be 1 since this is the density of normal distribution with variance=1/2)
- $\int_{-\infty}^{\infty} f(x)dx = \int_{-\infty}^{\infty} \frac{1}{\sqrt{\pi}} w(x)dx \approx \frac{1}{\sqrt{\pi}} \sum_{i=0}^{6} A_i \approx \frac{1}{\sqrt{\pi}} 1.772454 \approx 1{,}000000$

Adaptive quadrature and dimension of integrant

- Adaptive quadrature can introduce more points depending on the local behavior of f: in regions where the integral approximation is not yet stable (e.g. since f has a large change), more nodes might be added
- The R-function `integrate` uses adaptive Gaussian quadrature

- The algorithms discussed work in general well for one-dimensional cases
- For 2d or maybe 3d problems, they might be applied iteratively
- Curse of dimensionality: runtime growing exponentially with dimension

- For higher dimension, **Monte Carlo integration** often preferable

Monte Carlo estimator / MC integration

- In L6, we have generated $X_1, \ldots, X_n$ from a target distribution f
- A main use of these random draws is Monte Carlo integration: Calculate $\int f(x)dx$ or, more general, $\int h(x)f(x)dx$
- A Monte Carlo estimator of $\int h(x)f(x)dx$ is: $\boxed{\hat{\mu}_{MC} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)}$

- If h(x)=x, we estimate the distribution's mean with $\hat{\mu}_{MC} = \bar{X}$
- If h(x)=(x-$\bar{X}$)², we estimate the distribution's variance
- If h(x)=$\mathbf{1}$\{x>c\}, we estimate probability to be >c, e.g. a rejection probability: $\int_{-\infty}^{\infty} h(x)f(x)dx = \int_{c}^{\infty} f(x)dx = P(X > c)$ (see t-test simulation example in L6 and following example)

Example: Monte Carlo integration

- Background: Clinical study with two significance tests
- n_1 patients treated with high dose of a drug, n_2 with low dose, n_P with placebo; high dose compared to placebo (Z_1) and low dose compared to placebo (Z_2)

  Test 1: Reject H_{01} if $Z_1>c$

  Test 2: Reject H_{02} if $Z_2>c$
- Let Z_1 and Z_2 be standard normal distributed test statistics
- If c chosen conventionally, $c=1.96$ for $\alpha=0.025$, $P(Z_1>c)= P(Z_2>c)= 0.025$
- In this context, desired to control ***FamilyWise Error Rate*** (FWER) $P(Z_1>c \text{ or } Z_2>c)$ (reject any of the two)
- Z_1 and Z_2 are correlated

Example: Monte Carlo integration

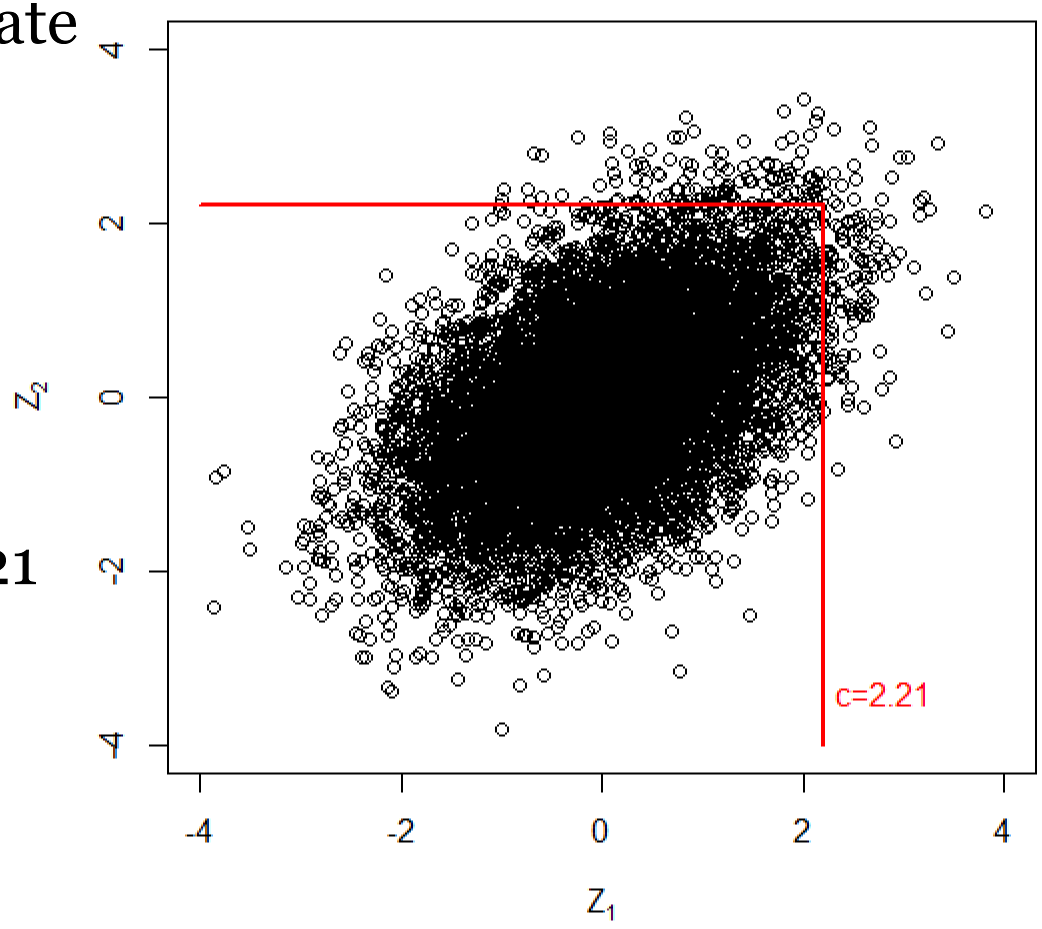
- We have $Z = \binom{Z_1}{Z_2} \sim N\left(\binom{0}{0}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\right)$ (multivariate normal) and want to determine c such that $P(Z_1 > c \text{ or } Z_2 > c) = \alpha$
- Sample from multivariate normal
- Determine Monte Carlo integral estimate for $P(Z_1 > c \text{ or } Z_2 > c)$ for arbitrary c
- Search then c such that $P(Z_1 > c \text{ or } Z_2 > c) = \alpha$ by bisection or sorting $\max(Z_1, Z_2)$ and taking 97.5%-percentile for $\alpha = 2.5\%$
- With $h(x_1, x_2) = \mathbf{1}\{x_1 > c \text{ or } x_2 > c\} = \mathbf{1}\{\max\{x_1, x_2\} > c\}$ we have

$$\boxed{\int_{\mathbb{R}^2} h(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x_1, x_2) f(x_1, x_2) dx_1\, dx_2 = P(Z_1 > c \text{ or } Z_2 > c)}$$

Example: Monte Carlo integration

- 10 000 random draws of bivariate normal with ρ=0.5
- For c=2.21 are 2.5% of draws upper and right to the red lines

- FWER is controlled at α=2.5%, if we reject any of H_{0i} for Z_i>2.21



Example: Monte Carlo integration

- R program to derive critical value based on Monte Carlo:

```
n   <- 1e+4
rho <- 0.5
x   <- matrix(rnorm(2*n), ncol = 2)
y   <- cbind(x[,1], rho * x[,1] + sqrt(1-rho^2) * x[,2]) #Multiv. normal
ym  <- apply(y, 1, max)    #Row-wise maximum
yms <- sort(ym)
cv  <- yms[round(n*0.975)] #Pick 97.5%-percentile in sample as critical value
cv
```

- Function **qmvnorm** in package **mvtnorm** can calculate/ simulate this value, too:

```
library(mvtnorm)

qmvnorm(0.975, tail = "lower.tail",
        corr = matrix(c(1,0.5,0.5,1), ncol = 2))
```

Importance sampling

- A Monte Carlo estimator of $\int h(x)f(x)dx$ is
$$\hat{\mu}_{MC} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)$$
- Depending on h, not all X_i equally relevant for this estimate
- We might want to focus more on certain X_i and with this derive an alternative Monte Carlo based estimator with reduced variance
- Idea:
  - Since $\int h(x)f(x)dx = \int h(x)\frac{f(x)}{g(x)}g(x)dx$, sample according to another density g which focuses on the **important** part of the sampling region
  - Correct estimate by weighting according to $\frac{f(x)}{g(x)}$

Importance sampling

- A Monte Carlo estimator of $\int h(x)f(x)dx = \int h(x)\frac{f(x)}{g(x)}g(x)dx$ is
$$\hat{\mu}_{MC} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)$$
- Importance sampling:
  - Choose g focusing on important regions (aiming for $g > f$ there, elsewhere $g < f$)
  - Sample according to g
  - Calculate $\hat{\mu}_{IS}^{*} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)w^{*}(X_i)$ with weights $w^{*}(X_i) = \frac{f(X_i)}{g(X_i)}$
- Important that it is possible to evaluate f and g and easy to sample from g

Importance sampling

- $\hat{\mu}_{IS}^* = \frac{1}{n}\sum_{i=1}^{n} h(X_i)w^*(X_i)$ with weights $w^*(X_i) = \frac{f(X_i)}{g(X_i)}$
  ($\hat{\mu}_{IS}^*$ is the sample mean of $t(X_i) = h(X_i)w^*(X_i), i = 1, \dots, n$)

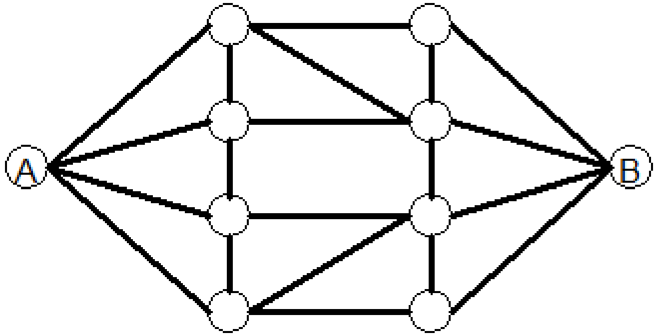
- $\hat{\mu}_{IS}^*$ is an unbiased estimator of $\mu = \int h(x)f(x)dx$

- The variance of $\hat{\mu}_{IS}^*$ is $\frac{\sigma_{IS*}^2}{n}$ with $\sigma_{IS*}^2 = \int (h(x)w^*(x) - \mu)^2 g(x)\, dx$
  (see Givens and Hoeting or Owen, Theorem 9.1)
  ➔ an estimator for variance of $\hat{\mu}_{IS}^*$ is $\frac{1}{n}$ times sample variance of

$$t(X_i) = h(X_i)w^*(X_i), i = 1, \dots, n$$

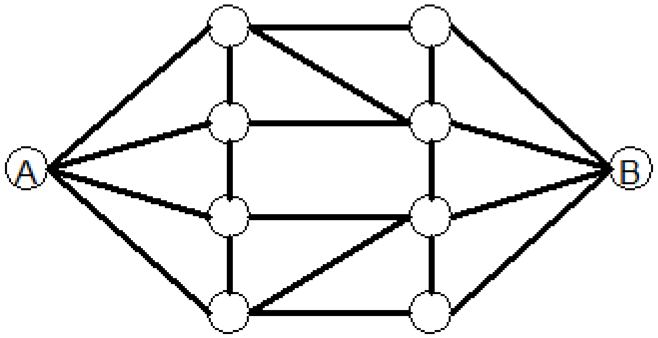
Importance sampling – network analysis

- Network analysis: failure probabilities can be extremely small ➔ Importance sampling can be useful (Givens and Hoeting, example 6.9):
- A network consists of nodes and edges (visualized by circles and lines)



- Each edge is intact with high probability but has a failure probability p_i which typically is small
- Whole network intact if endnode B reachable from startnode A via intact edges, broken otherwise

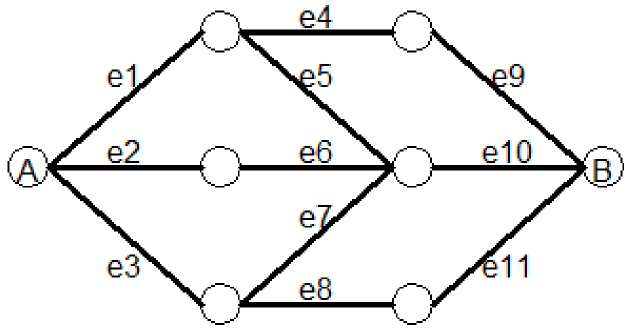
Importance sampling – network analysis



- Run n times:
  - Simulate each edge (if intact or broken)
  - Compute whether network intact or broken
- Problem: Only a few networks will be broken
- To decrease variance of estimator, simulate with failure-probabilities $p_i^* > p_i$ and use $\hat{\mu}_{IS}^*$

Importance sampling – network example

- Example:



- Assume that p_i=0.05 for all edges
- A function **`net`** computes if the network is intact (**`net(x)=1`**) or broken (**`net(x)=0`**) for vector of edge-states **x**=(x_1,…,x_{11})
- To decrease variance of estimator, simulate with failure-probabilities $p_i^*>p_i$ and use $\hat{\mu}_{IS}^*$
- We use here p_i^*=0.3

Importance sampling – network example

```
sim        <- 100000
totaledges <- 11
p          <- 0.05
ps         <- 0.3
simmat     <- matrix(rbinom(sim*totaledges, size=1, prob=1-ps), ncol=totaledges)
broken     <- 1-apply(simmat, 1, net)
nbrokenedg <- totaledges - rowSums(simmat)
w <- dbinom(nbrokenedg, size=totaledges, prob=p) /
     dbinom(nbrokenedg, size=totaledges, prob=ps)
#The following formula gives same importance weights ((6.48) in GH, 2013):
#w2 <- ((1-p)/(1-ps))^totaledges * (p*(1-ps)/(ps*(1-p)))^nbrokenedg
bhatIS <- mean(broken*w)
```

- We get here an estimate $\hat{\mu}_{IS}^{*} = 0.000781$
- sd is 0.0000165 obtained by `sqrt(var(broken*w)/sim)`
- sd is lower by factor 5.9 compared to standard Monte Carlo estimate based on same number of simulations

Antithetic sampling

- Given a Monte Carlo estimator $\hat{\mu}_{MC1} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)$, there might be another $\hat{\mu}_{MC2}$ which has same distribution and is negatively correlated (let $\rho = \mathrm{Corr}(\hat{\mu}_{MC1}, \hat{\mu}_{MC2})$)
- Then, $\hat{\mu}_{AS} = (\hat{\mu}_{MC1} + \hat{\mu}_{MC2})/2$ is an estimator for same target variable and has lower variance (factor $\frac{1+\rho}{2}$ lower)

- Example: Let X be a symmetric random var. with mean 0.
  Interest in calculating p=P(X>1) by Monte Carlo sim.
- Use $\hat{\mu}_{MC1} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)$ with $h(X_i) = \mathbf{1}\{X_i > 1\}$
- The same distribution has $\hat{\mu}_{MC2} = \frac{1}{n}\sum_{i=1}^{n} \tilde{h}(X_i)$ with $\tilde{h}(X_i) = \mathbf{1}\{X_i < -1\}$ (due to symmetry) and they are negatively correlated, $\rho = -p/(1-p)$

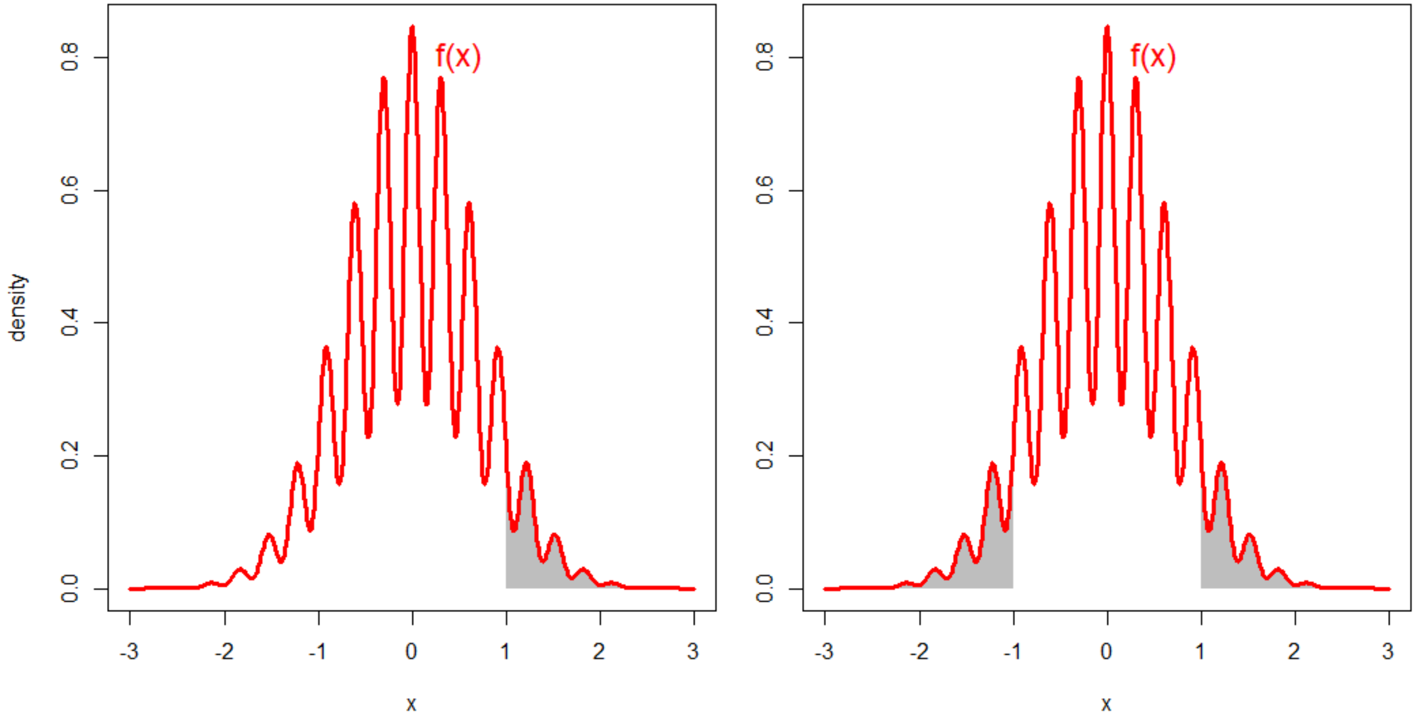
Importance and antithetic sampling – an example

- Example: Let X be a symmetric random var. with complicated density *f* and calculate *p*=P(X>1) by Monte Carlo simulation

- Use $\hat{\mu}_{MC1} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)$ with $h(X_i) = \mathbf{1}\{X_i > 1\}$

- $\hat{\mu}_{MC2} = \frac{1}{n}\sum_{i=1}^{n} \tilde{h}(X_i)$ with $\tilde{h}(X_i) = \mathbf{1}\{X_i < -1\}$ has the same distribution

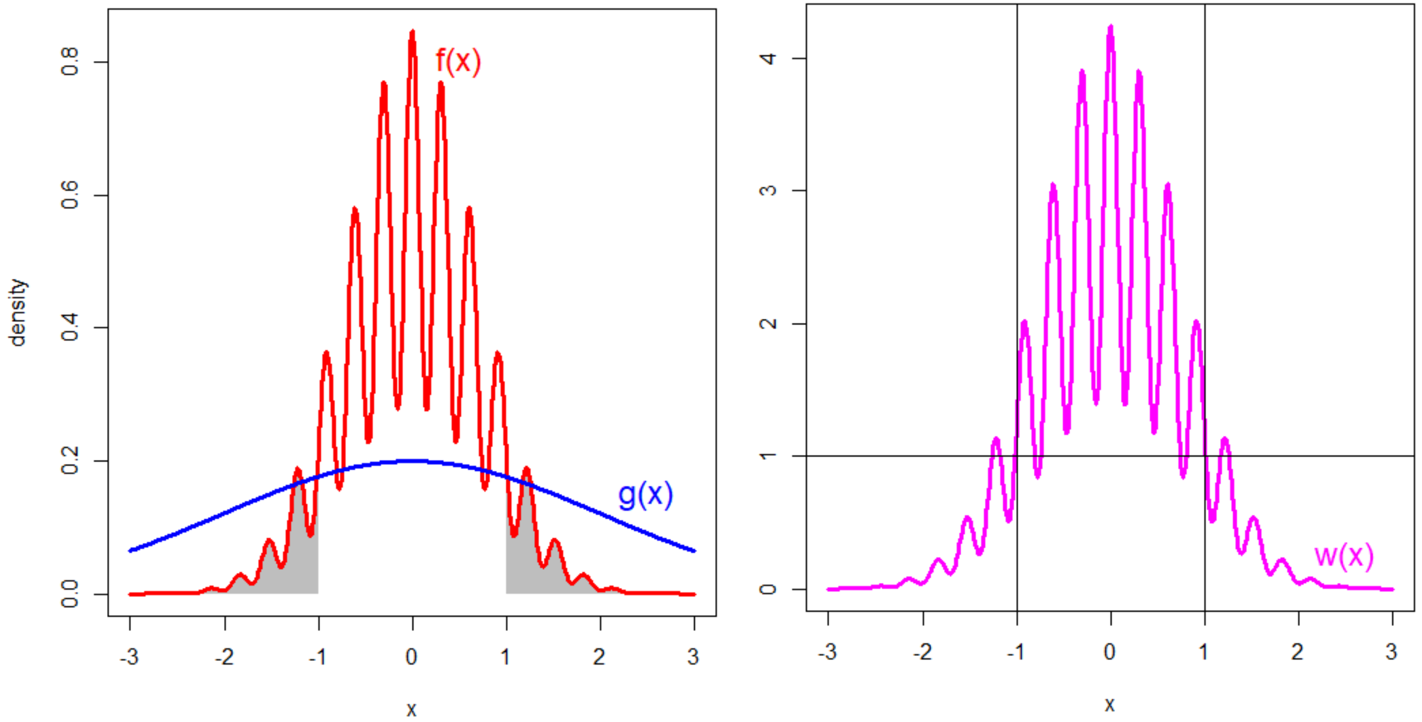
- So, we compute 2*p*=P(|X|>1) and will use importance sampling for it



Importance and antithetic sampling – an example

- For importance sampling, we want to oversample important regions and undersample otherwise
- We use here a normal distribution with standarddeviation 2 as sampling distribution g
- The weight is then $w=f/g$

```
f <- function(t){
  ct <- (2+cos(t*(64/pi)))
  exp(-t^2)*ct/3.544909
}
sim <- 1000000
y <- rnorm(sim,sd=2)
w <- f(y)/dnorm(y,sd=2)
z <- (abs(y)>1)*w
mean(z)/2
[1] 0.07368936
```



Importance and antithetic sampling – an example

```
z <- (abs(y)>1)*w
p <- mean(z)/2
p
[1] 0.07368936
```

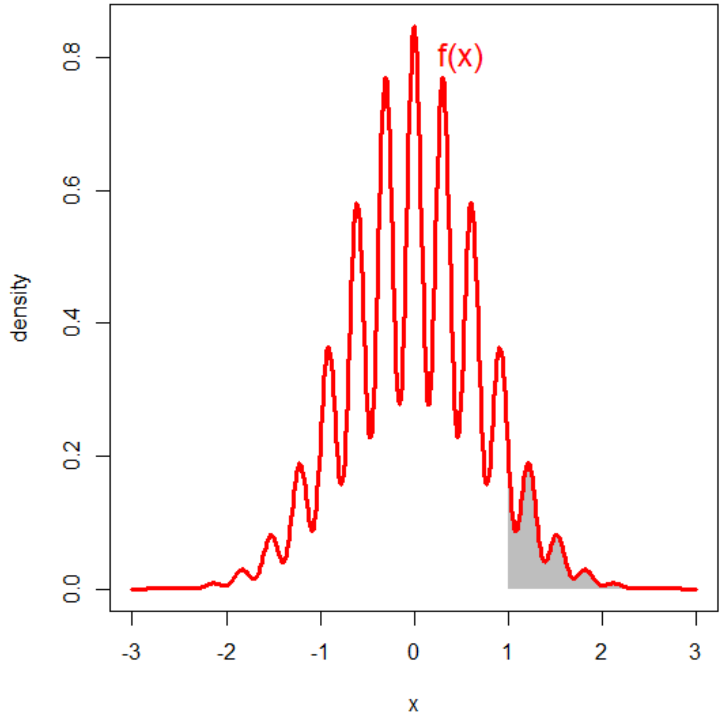
- What is the uncertainty in this estimate?
- sd for the IS estimate of p:

```
sdIS <- sqrt(var((y>1)*w)/sim)
sdIS
[1] 0.000210754
```

- sd for the AS estimate of p:

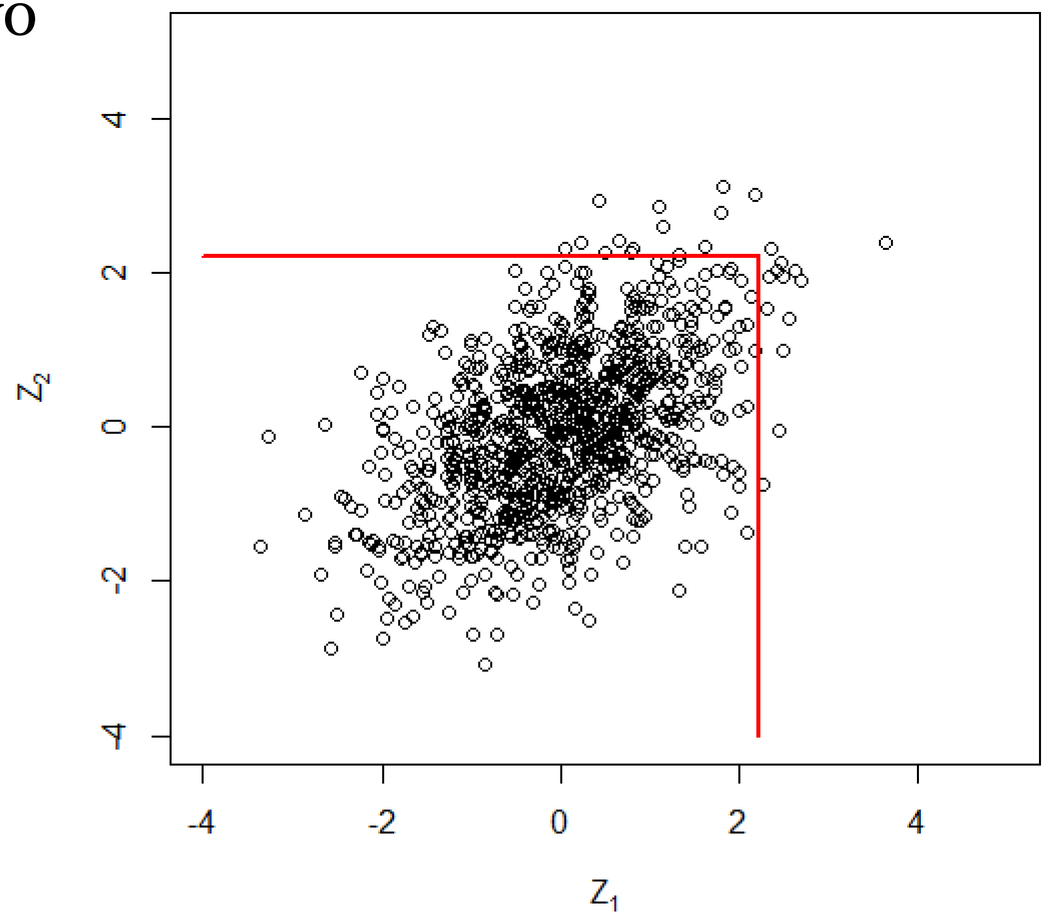
```
rho <- -p/(1-p)
sd  <- sdIS*(1+rho)/2
sd
[1] 9.699411e-05
```

- 95% CI for p: (0.07350, 0.07388)



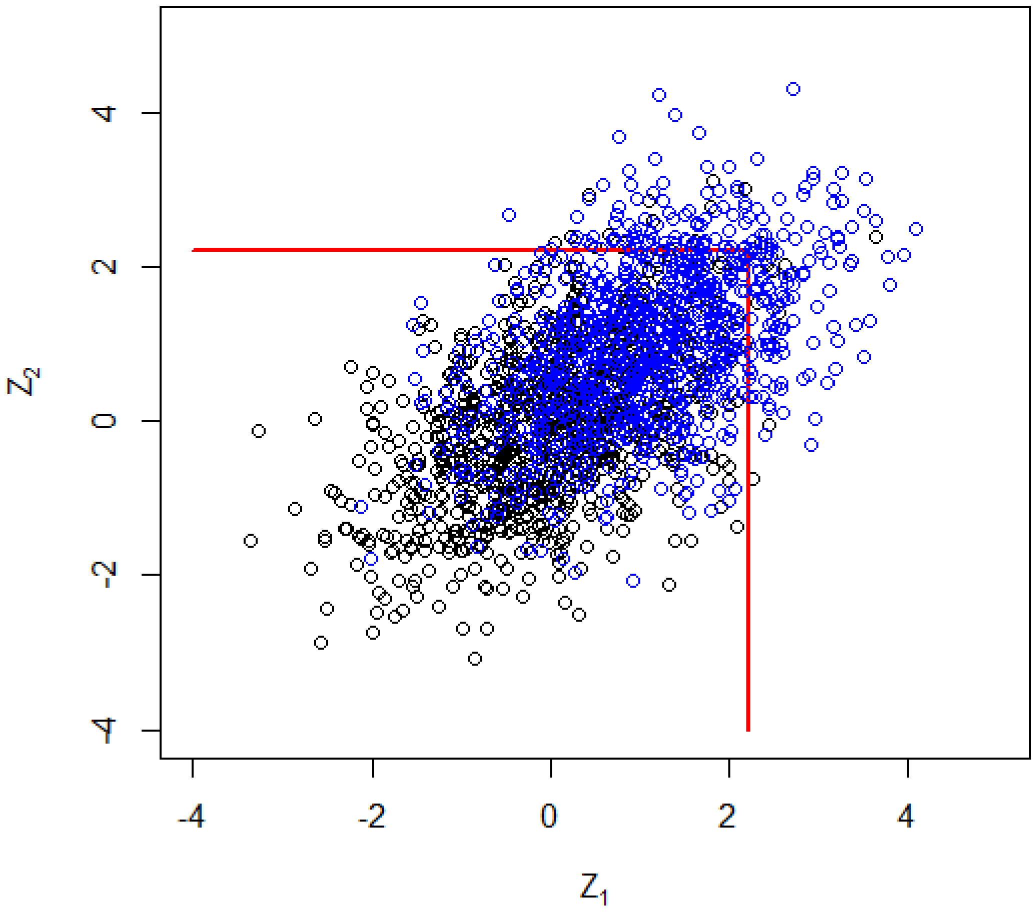
Ex.: MC integration with importance sampling

- Going back to example with two significance tests
- We fix now c=2.21
- We are interested to compute $P(Z_1>c \text{ or } Z_2>c)$ with high precision using importance sampling
- Which importance functions *g* would be good?



Ex.: MC integration with importance sampling

- For illustration we use $N\left(\begin{pmatrix}\delta \\ \delta\end{pmatrix}, \begin{pmatrix}1 & 0.5 \\ 0.5 & 1\end{pmatrix}\right)$ with $\delta=1$ for g (might be better choices, too)
- Draws in lower-left corner:
  - less often sampled
  - overweighted if sampled
  - have lower precision (but $h=0$ there, so low precision is no problem)



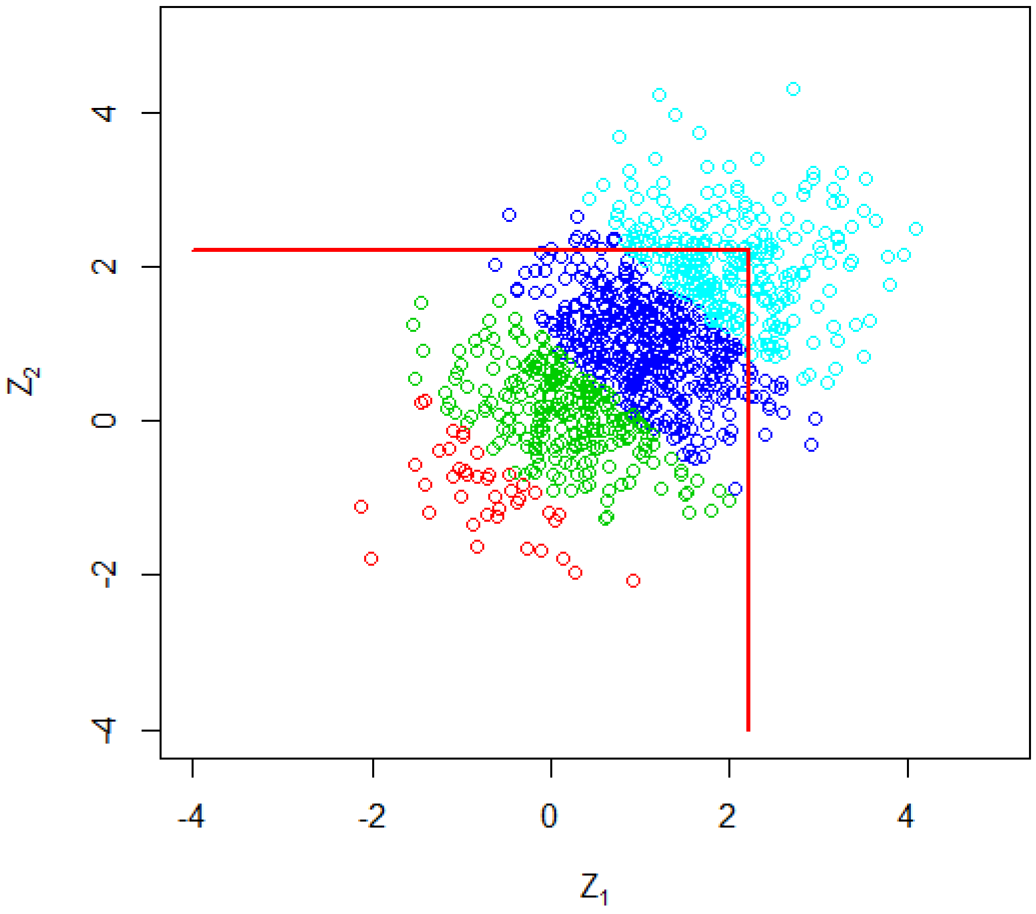
Ex.: MC integration with importance sampling

- Standard deviation

n	1000	100 000
$\hat{\mu}_{MC}$	0.0050	0.00049
$\hat{\mu}^*_{IS}$	0.0020	0.00020

- n=100 000, $\hat{\mu}^*_{IS} = 0.02489$

- Draws with weights above 4, in [1,4], in [0.25,1), and below 0.25, respectively are in different colors in picture



Importance sampling with standardized weights

- Importance sampling estimator with unstandardized weights of $\int h(x)f(x)dx = \int h(x)\frac{f(x)}{g(x)}g(x)dx$ is

$$\hat{\mu}_{IS}^{*} = \frac{1}{n}\sum_{i=1}^{n} h(X_i)w^*(X_i) \text{ with weights } w^*(X_i) = \frac{f(X_i)}{g(X_i)}$$

- Importance sampling estimator **with standardized weights** is

$$\hat{\mu}_{IS} = \sum_{i=1}^{n} h(X_i)w(X_i) \text{ with } w^*(X_i) = \frac{f(X_i)}{g(X_i)},\ w(X_i) = \frac{w^*(X_i)}{\sum_{j=1}^{n} w^*(X_j)}$$

- $\hat{\mu}_{IS}$ can be used if f known up to proportionality constant
- $\hat{\mu}_{IS}$ has a slight bias and variance more complicated

Importance sampling with standardized weights

- Importance sampling estimator **with standardized weights** is $\hat{\mu}_{IS} = \sum_{i=1}^{n} h(X_i)w(X_i)$ with $w^*(X_i) = \frac{f(X_i)}{g(X_i)}$, $w(X_i) = \frac{w^*(X_i)}{\sum_{j=1}^{n} w^*(X_j)}$
- $\hat{\mu}_{IS}$ has a slight bias,

$$\mathrm{E}(\hat{\mu}_{IS} - \mu) = \frac{1}{n}\left[\mu \mathrm{Var}\left(w^*(X)\right) - \mathrm{Cov}\left(t(X), w^*(X)\right)\right] + O\left(\frac{1}{n^2}\right).$$

- Its variance is $\mathrm{Var}(\hat{\mu}_{IS}) = \frac{1}{n}\left[\mathrm{Var}\left(t(X)\right) + \mu^2 \mathrm{Var}\left(w^*(X)\right) - 2\mu \mathrm{Cov}\left(t(X), w^*(X)\right)\right] + O(1/n^2).$
- To estimate these quantities, one can use the sample statistics for $w^*(X)$ and $t(X) = h(X)w^*(X)$ and replace μ by its estimate