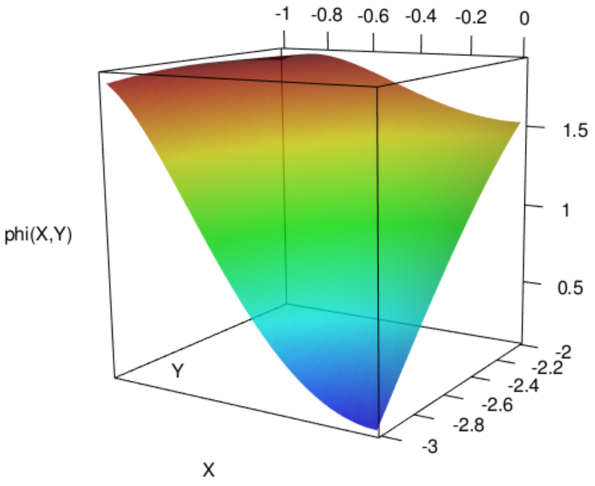
Monte Carlo Integration

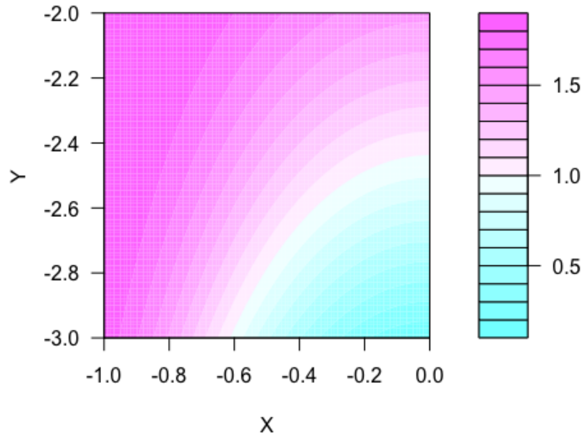
Viktoriia Oliinyk, Maksim Berezov

Setup

We consider the function $\varphi(x,y) = 0.15 + 1.75 sin\Big(cos(x)\sqrt{1+x^2+y^2}\Big)$ on the support $S = [-1,0] \times [-3,-2]$.

3D plot of function $\varphi(x,y)$ and its contourplot





Notice that, as required, this function

- cannot be presented as multiplication of two functions $\varphi_1(x), \varphi_2(y)$
- is positive on S
- is bounded on S.

We use 4 approaches to calculate the integral $\int_{-3}^{-2}\int_{-1}^{0}\varphi(x,y)dxdy$.

Standard numeric calculation (using the Gaussian quadrature) yields the value of integral 1.335642.

1. Naïve Monte Carlo

We simulate samples of $n = 10^3$ i.i.d. points $X_i \sim U_{[-1,0]}, Y_i \sim U_{[-3,-2]}, W_i \sim U_{[0,K]}$, where $K = \max_{(x,y)\in S}\varphi(x,y) = 1.9$.

We estimate integral as $I_1 = \frac{K \cdot Vol(S)}{n}\sum_{i=1}^{n} 1(\varphi(X_i,Y_i) > W_i)$.

```
I1 <- (b-a)*(d-c)*K*sum(phi(X,Y) > W)/n
```

2. Basic Monte Carlo

We simulate samples of $n = 10^3$ i.i.d. points $X_i \sim U_{[-1,0]}, Y_i \sim U_{[-3,-2]}$.

We estimate integral as $I_2 = \frac{Vol(S)}{n} \cdot \sum_{i=1}^{n}\varphi(X_i,Y_i)$.

```
I2 <- (b-a)*(d-c)*mean(phi(X,Y))
```

3. Importance Sampling

a) X and Y are independent

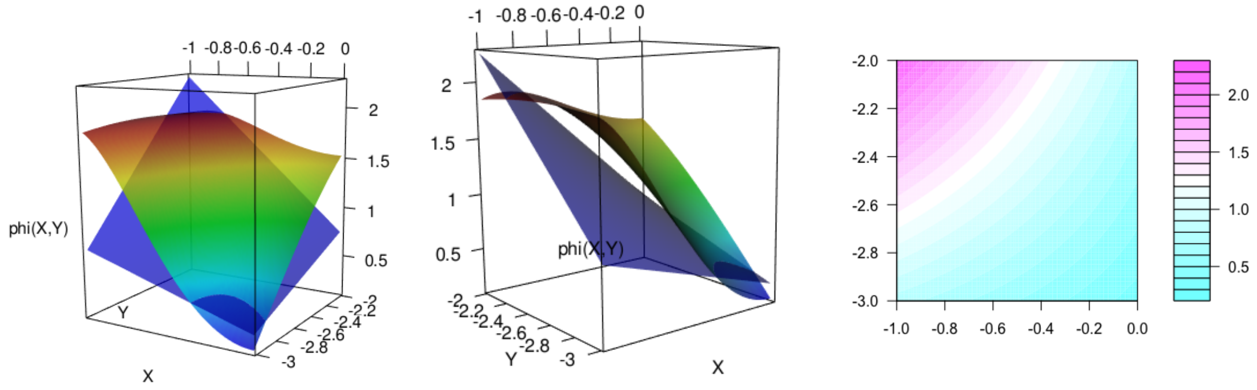
Now, we suppose that $X_i, Y_i, i \in \{1, ..., n\}$ are independent but not uniform on $[-1, 0]$ and $[-3, -2]$, respectively.

We suggest $f(x) = (-x + \frac{1}{2}) \cdot 1(x)_{[-1,0]}$ to be the density of X and $g(y) = (y + 3.5) \cdot 1(y)_{[-3,-2]}$ to be the density of Y.

We make sure $\int_{-1}^{0} f(x)dx = \int_{-3}^{-2} g(y)dy = 1$.

Next, the couple (X, Y) has joint distribution $p(x, y) = f(x) \cdot g(y) = (-x + \frac{1}{2}) \cdot (y + 3.5) \cdot 1(x, y)_{[-1,0]\times[-3,-2]}$, which verifies $\iint_S p(x, y)dS = 1$.

3D plots of the joint distribution $p(x, y)$ and its contourplot



Remark: Notice that $p(x, y)$ (in blue) is a plot of joint density, hence it must integrate to 1. It is important that the ratio between $\varphi(x, y)$ and $p(x, y)$ is close to constant value, so their scales may differ.

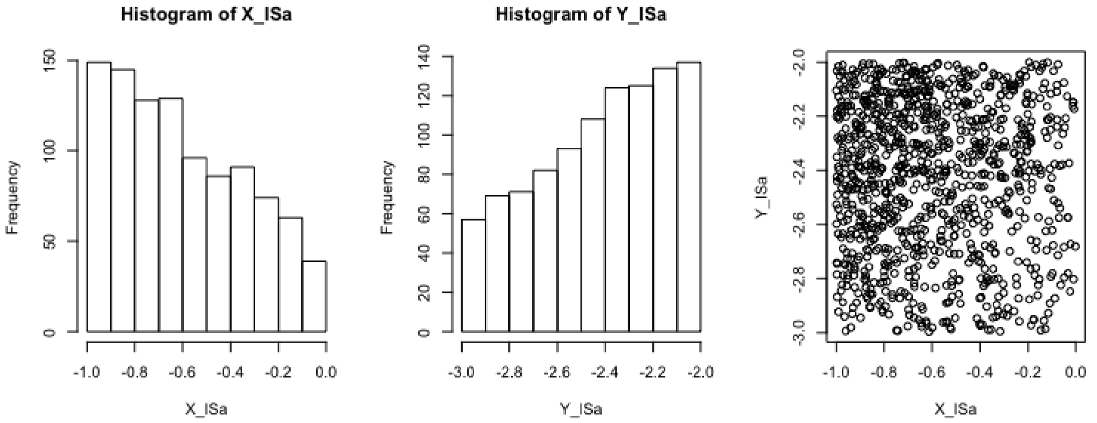
We simulate sample of $n = 10^3$ couples $(X, Y) \sim p$ using *inversion method*.

To use inversion method, we calculated c.d.f. and their inverses, respectively:

$F(x) = 0.5(-x^2 + x + 2)$, $F^{-1}(u) = 0.5(1 - \sqrt{9 - 8u})$

$G(y) = 0.5(y^2 + 7y + 12)$, $G^{-1}(v) = -3.5 + 0.5\sqrt{8v + 1}$.

We check that simulated couples have correct distribution by plotting histograms of marginal distributions and scatter plot of joint distribution.



We estimate integral as $I_{3a} = \frac{1}{n}\sum_{i=1}^{n}\frac{\varphi(X_i,Y_i)}{p(X_i,Y_i)}$.

```
U <- runif(n,0,1)
V <- runif(n,0,1)
X_ISa <- Finv_u(U)
Y_ISa <- Ginv_v(V)
I3a <- 1/n*sum(phi(X_ISa,Y_ISa)/p(X_ISa,Y_ISa))
```

b) X and Y are dependent

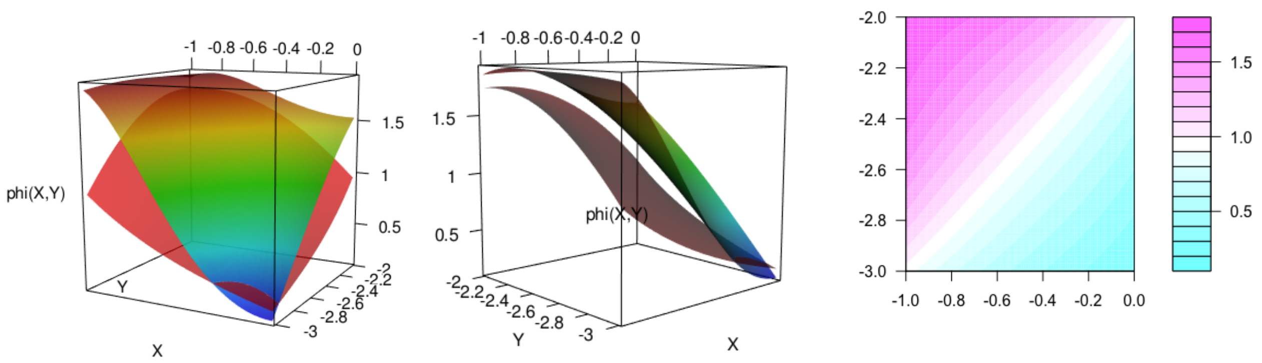
Finally, we add dependence to the distribution of (X, Y) using Frank copula. We used Frank copula with $\theta = 0.5$ to make sure that couples where X and Y are both large or both low would occur more frequently.

We preserve the marginal distributions of X and Y to be the same as in the previous paragraph, $f(x) = (-x + \frac{1}{2}) \cdot 1(x)_{[-1,0]}, g(y) = (y + 3.5) \cdot 1(y)_{[-3,-2]}$.

We set the joint distribution $p_{Fr}(x, y) = f(x)g(y)c(F(x), G(y))$,

where $c(u, v) = \frac{\partial^2 C(u,v)}{\partial u \partial v} = \frac{e^{-\theta u}(-\theta e^{-\theta v})(e^{-\theta}-1)}{(e^{-\theta}-1+(e^{-\theta u}-1)(e^{-\theta v}-1))^2}$.

3D plots of the joint distribution $p_{Fr}(x, y)$ and its contourplot



The function $\varphi(x, y)$ and the joint density $p_{Fr}(x, y)$ look very similar. That is good.

We generate $n = 10^3$ couples $(X, Y) \sim p_{Fr}$ as follows.

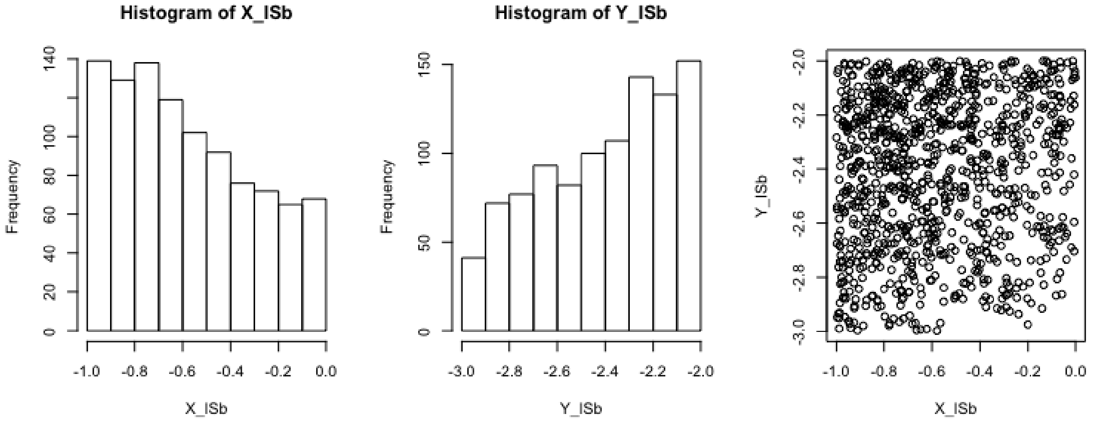
We generate n i.i.d. couples $(U_i, V_i) \sim C$, and we set $(X_i, Y_i) = (F^{-1}(U_i), G^{-1}(V_i))$.

In order to simulate $(U_i, V_i) \sim C$, we calculated the inverse function of $F_u(z) = \frac{\partial C(u,v)}{\partial u}$:

$F_u^{-1}(z) = -\frac{1}{\theta} log\Big(1 + \frac{e^{\theta u} z(e^{-\theta}-1)}{1-(1-e^{\theta u})z}\Big)$.

```
U <- runif(n)
Z <- runif(n)
V <- dC_inv(U,Z)
X_ISb <- Finv_u(U)
Y_ISb <- Ginv_v(V)
I3b <- 1/n*sum(phi(X_ISb,Y_ISb)/p_fr(X_ISb,Y_ISb))
```

We check that simulated couples have correct distribution by plotting histograms of marginal distributions and scatter plot of joint distribution.

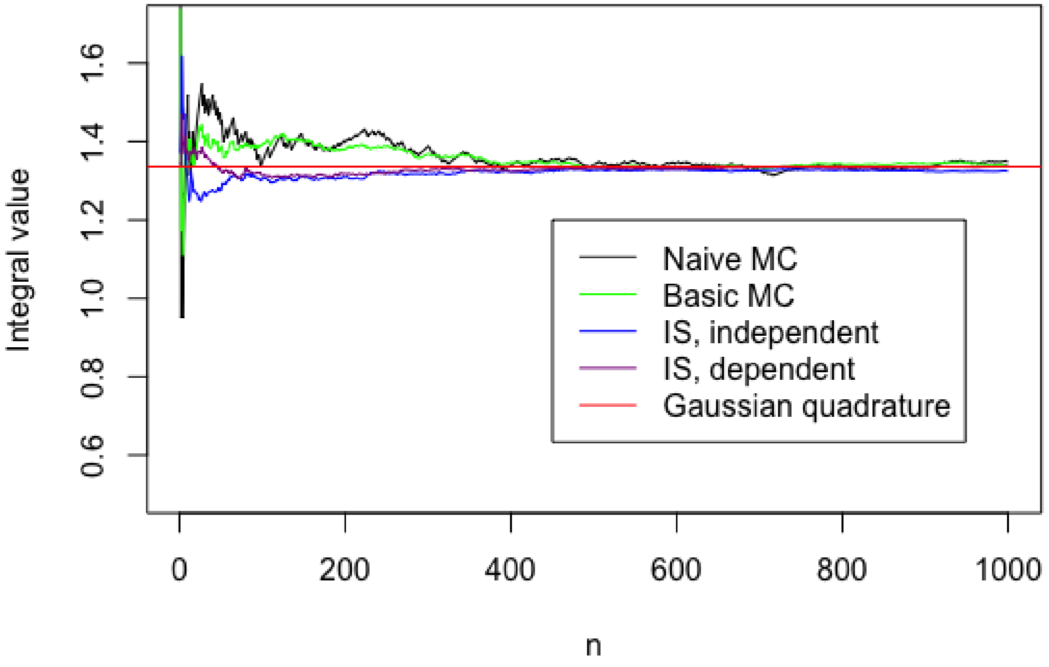


We estimate integral as $I_{3b} = \frac{1}{n}\sum_{i=1}^{n} \frac{\varphi(X_i, Y_i)}{p_{Fr}(X_i, Y_i)}$.

Convergence Study

Finally, we compare the convergence of discussed methods as $n \to \infty$.

We plot the values of integral calculated using different methods as functions of n.



As we see, Naïve Monte Carlo is naïve indeed and behaves the worst.

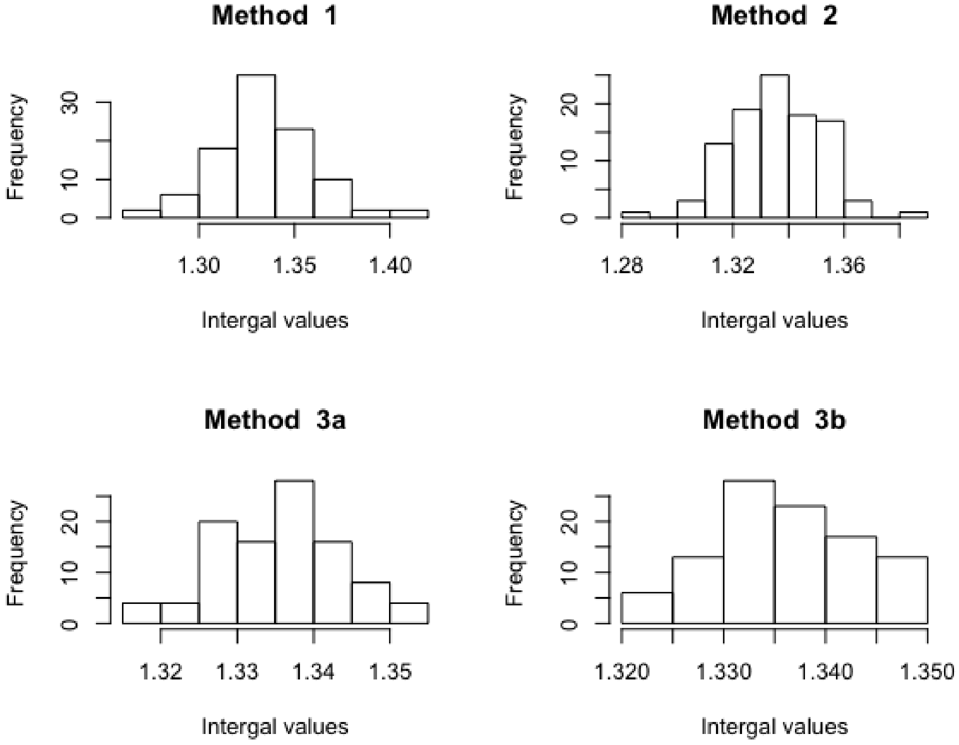
Integral values converge a bit better when we use Basic Monte Carlo. The best convergence of integral values is attained when Importance Sampling is used. Moreover, when (X, Y) are dependent, the convergence is the fastest.

Robustness Study

Now, we iterate each of the methods described above 100 times to see how the integral values obtained are distributed.

We print means, standrad deviations and plot histogams of integrals obtained through each method.

	Method name	Mean	St.Dev.	abs(target - $\hat{I}$)
Method 1	Naïve MC	1.337885	0.026632024	0.0022426362
Method 2	Basic MC	1.334655	0.013619909	0.0009873642
Method 3a	IS. X and Y independent	1.335867	0.009556707	0.0002246358
Method 3b	IS. X and Y dependent	1.335776	0.006771843	0.0001336358



Reminder: The value of integral calculated by R is 1.335642.

Again, the results are ordered from the worst to the best, where the closer the mean is to the target value and the lower the variance is, the better.

Conclusion

The Naïve Monte Carlo method converges slowly to the integral value and is not very stable over iterations.

The Basic Monte Carlo converges a bit faster but again is not very stable over iterations.

The Importance Sampling gives the best results in terms of convergence rate and robustness. Moreover, its version with dependend (X, Y) with joint distribution imitating the shape of the function provides the best results.