Programmieraufgabe 3.

zu bearbeiten bis **Montag, 28.04.2014**.

In der Praxis kommen oftmals partielle Differentialgleichungen mit *log-normal verteilten* Diffusionskoeffizienten vor. Gesucht ist dann die Lösung von

$$\left.\begin{aligned} -\operatorname{div}\big(\alpha(\mathbf{x},\mathbf{y})\nabla u(\mathbf{x},\mathbf{y})\big) &= f(\mathbf{x}), \quad && \mathbf{x}\in D \\ u(\mathbf{x},\mathbf{y}) &= 0, && \mathbf{x}\in\Gamma := \partial D \end{aligned}\right\}\quad \mathbf{y}\in\mathbb{R}^M,$$

wobei der Diffusionskoeffizient gegeben ist durch

$$\alpha(\mathbf{x},\mathbf{y}) = \exp\left(\sum_{i=1}^{M}\psi_i(\mathbf{x})y_i\right) \tag{1}$$

mit unabhängigen und standardnormalverteilten Zufallsgrössen y_i. Wir sind daran interessiert, die Statistika der Lösung u, wie Erwartungswert und Varianz, zu bestimmen. Dabei können die bereits bekannten Sampling-Verfahren verwendet werden. Allerdings muss hierzu das Integrationsgebiet entsprechend transformiert werden. Es gilt beispielsweise für den Erwartungswert der Lösung

$$\mathbb{E}[u](\mathbf{x}) = \frac{1}{(\sqrt{2\pi})^M}\int_{\mathbb{R}^M} u(\mathbf{x},\mathbf{y})e^{-\frac{\mathbf{y}^\intercal\mathbf{y}}{2}}\,\mathrm{d}\mathbf{y} = \int_{(0,1)^M} u(\mathbf{x},\boldsymbol{\Phi}^{-1}(\mathbf{z}))\,\mathrm{d}\mathbf{z}.$$

Hierbei bezeichnet $\Phi^{-1}\colon (0,1)\to\mathbb{R}$ die *inverse Verteilungsfunktion* zur Normalverteilung und es gilt $\boldsymbol{\Phi}^{-1}(\mathbf{z}) = [\Phi^{-1}(z_1),\ldots,\Phi^{-1}(z_M)]^\intercal$. Die inverse Verteilungsfunktion der Normalverteilung kann sehr effizient mit dem *Moro-Verfahren* implementiert werden. Hierbei handelt es sich um eine stückweise rationale Interpolation der inversen Normalverteilung:

Input : $z\in(0,1)$
Output : $Ninv = \Phi^{-1}(z)$
$p = z - 0.5$
if $|p| < 0.42$ **then**
  $r = p*p$
  $Ninv = p*(((E3*r+E2)*r+E1)*r+E0)/((((F3*r+F2)*r+F1)*r+F0)*r+1)$
else
  if $p<0$ **then**
    $r = z$
  else
    $r = 1-z$
  end
  $r = \log(-\log(r))$
  $r = G0+r*(G1+r*(G2+r*(G3+r*(G4+r*(G5+r*(G6+r*(G7+r*G8)))))))$
  if $p<0$ **then**
    $Ninv = -r$
  else
    $Ninv = r$
  end
end

Die im Algorithmus auftretenden Konstanten sind gegeben durch

$E0 = 2.50662823884$	$E1 = -18.61500062529$	$E2 = 41.39119773534$
$E3 = -25.44106049637$	$F0 = -8.47351093090$	$F1 = 23.08336743743$
$F2 = -21.06224101826$	$F3 = 3.13082909833$	$G0 = 0.3374754822726147$
$G1 = 0.9761690190917186$	$G2 = 0.1607979714918209$	$G3 = 0.0276438810333863$
$G4 = 0.0038405729373609$	$G5 = 0.0003951896511919$	$G6 = 0.0000321767881768$
$G7 = 0.0000002888167364$	$G8 = 0.0000003960315187$	

Aufgabe 1. Schreiben Sie eine matrixwertig auswertbare Funktion

```
function Ninv=moro(Y),
```

die das Moro-Verfahren numerisch umsetzt. Testen Sie Ihre Implementierung anhand der Funktion `testmoro.m` von der Webseite. Diese Funktion generiert einen Vektor von uniform verteilten Zufallszahlen und wendet die Funktion `moro` darauf an. Nun wird gezählt, wie oft eine Zufallszahl innerhalb eines bestimmten Intervalls einer äquidistanten Unterteilung der x-Achse liegt. Nach entsprechender Normierung approximieren diese Werte die Dichte

$$\varphi(x) = \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)$$

der Normalverteilung.

Aufgabe 2. Approximieren Sie den Erwartungswert des Poisson-Problems mit lognormalem Koeffizienten (1) auf dem Einheitsquadrat. Verwenden Sie hierzu das Monte-Carlo- und das Quasi-Monte-Carlo-Verfahren mit $N = 10^4$ Samples. Dabei seien $\psi_i(\mathbf{x}) = 2^{-i} \sin(i\pi x_1) \sin(i\pi x_2)$ und die rechte Seite $f(\mathbf{x}) = 1$ gegeben. Die Dimension des Parameterbereichs sei $M = 5$. Das Grobgitter auf dem Einheitsquadrat sollte 5 mal verfeinert werden (Level 5). Verwenden Sie die Lösung der Quasi Monte-Carlo Approximation als Referenz, um damit die Konvergenz der Monte Carlo Approximation zu untersuchen. Bestimmen Sie hierzu den relativen L^2-Fehler der Monte-Carlo Lösungen mit $10, 100, 1000, 10000$ Samples und tragen diesen in einem `loglog`-Plot gegen die Anzahl der Samples auf.