1. 蒙特卡洛方法收敛性

对于 $f: R^{N_\theta} \rightarrow R$，定义 $\mathrm{Var}_\rho[f] = \mathbb{E}_\rho[(f - \mathbb{E}_\rho[f])^2]$。我们有

$$\begin{aligned}\mathbb{E}\Big[\frac{1}{J}\sum_{j=1}^{J} f(\theta^j) - \mathbb{E}[f]\Big] = 0\\ \mathbb{E}\Big[\big(\frac{1}{J}\sum_{j=1}^{J} f(\theta^j) - \mathbb{E}[f]\big)^2\Big] = \frac{\mathrm{Var}_\rho[f]}{J}\end{aligned}$$

证明

由于$\theta^j \sim \rho$， 我们有

$$\mathbb{E}\Big[\frac{1}{J}\sum_{j=1}^{J} f(\theta^j) - \mathbb{E}[f]\Big] = \frac{1}{J}\sum_{j=1}^{J}\mathbb{E}[f] - \mathbb{E}[f] = 0$$

对于方差，我们有

$$\begin{aligned}\mathbb{E}\Big[\big(\frac{1}{J}\sum_{j=1}^{J} f(\theta^j) - \mathbb{E}[f]\big)^2\Big] &= \mathbb{E}\Big[\frac{1}{J}\sum_{j=1}^{J}\big(f(\theta^j) - \mathbb{E}[f]\big)^2\Big]\\ &= \frac{1}{J}\mathbb{E}\Big[\big(f(\theta) - \mathbb{E}[f]\big)^2\Big]\\ &= \frac{\mathrm{Var}_\rho[f]}{J}\end{aligned}$$

2. 重要性采样方法收敛性

定义$L(\theta) = e^{-\Phi(\theta)}$，$\rho^* = \frac{1}{Z}L(\theta)\rho(\theta)$， 我们将证明

$$\begin{aligned}\sup_{|f|_\infty \leq 1}\Bigg|\mathbb{E}_\rho\Big[\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)} - \mathbb{E}_{\rho^*}[f]\Big]\Bigg| \leq 2\frac{1+\chi^2[\rho^*\|\rho]}{J}\\ \sup_{|f|_\infty \leq 1}\mathbb{E}_\rho\Big[\Big(\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)} - \mathbb{E}_{\rho^*}[f]\Big)^2\Big] \leq 4\frac{1+\chi^2[\rho^*\|\rho]}{J}\end{aligned}$$

证明

我们有

$$\begin{aligned}&\rho^{*J}_{IS}(f) = \frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)}\\ &\chi^2[\rho^*\|\rho] = \int\frac{{\rho^*}^2}{\rho} - 1 = \frac{\mathbb{E}_\rho[L(\theta)^2]}{Z^2} - 1\\ &\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)} - \mathbb{E}_{\rho^*}[f] = \frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)}\big(1 - \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}\big) + \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}f(\theta^j) - \mathbb{E}_{\rho^*}[f]\end{aligned}$$

我们用 $\rho^{*J}_{IS}(f) \leq |f|_\infty$ 和 $\mathbb{E}_\rho\Big[\big(\frac{1}{J}\sum_{j=1}^J f(\theta^j) - \mathbb{E}[f]\big)^2\Big] \leq \mathbb{E}_\rho[f(\theta)^2]$，对于方差

$$\begin{aligned}\mathbb{E}_\rho\Big[\Big(\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)} - \mathbb{E}_{\rho^*}[f]\Big)^2\Big] &= \mathbb{E}_\rho\Big[\Big(\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)}\big(1 - \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}\big) + \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}f(\theta^j) - \mathbb{E}_{\rho^*}[f]\Big)^2\Big]\\ &\leq 2\mathbb{E}_\rho\Big[\Big(\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)}\Big)^2\big(1 - \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}\big)^2 + \Big(\frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}f(\theta^j) - \mathbb{E}_{\rho^*}[f]\Big)^2\Big]\\ &\leq 2\mathbb{E}_\rho\Big[\big(1 - \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}\big)^2 + \Big(\frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}f(\theta^j) - \mathbb{E}_{\rho^*}[f]\Big)^2\Big]\\ &\leq \frac{2}{J}\Big(\mathbb{E}_\rho\Big[\frac{L(\theta)^2}{Z^2}\Big] + \mathbb{E}_\rho\Big[\frac{L(\theta)^2 f(\theta)^2}{Z^2}\Big]\Big)\\ &\leq 4\frac{\chi^2[\rho^*\|\rho] + 1}{J}\end{aligned}$$

对于偏差

$$\begin{aligned}\mathbb{E}_\rho\Big[\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)} - \mathbb{E}_{\rho^*}[f]\Big] &= \mathbb{E}_\rho\Big[\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)}\big(1 - \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}\big) + \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}f(\theta^j) - \mathbb{E}_{\rho^*}[f]\Big]\\ &= \mathbb{E}_\rho\Big[\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)}\big(1 - \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}\big)\Big]\\ &= \mathbb{E}_\rho\Big[\big(\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)} - \mathbb{E}_{\rho^*}[f]\big)\big(1 - \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}\big)\Big]\\ &\leq \sqrt{\mathbb{E}_\rho\Big[\big(\frac{\sum_{j=1}^J L(\theta^j)f(\theta^j)}{\sum_{j=1}^J L(\theta^j)} - \mathbb{E}_{\rho^*}[f]\big)^2\Big]}\sqrt{\mathbb{E}_\rho\Big[\big(1 - \frac{1}{J}\sum_{j=1}^J\frac{L(\theta^j)}{Z}\big)^2\Big]}\\ &\leq \sqrt{4\frac{\chi^2[\rho^*\|\rho] + 1}{J}}\sqrt{\frac{1}{J}\mathbb{E}_\rho\Big[\frac{L(\theta^j)^2}{Z^2}\Big]}\\ &= 2\frac{\chi^2[\rho^*\|\rho] + 1}{J}\end{aligned}$$

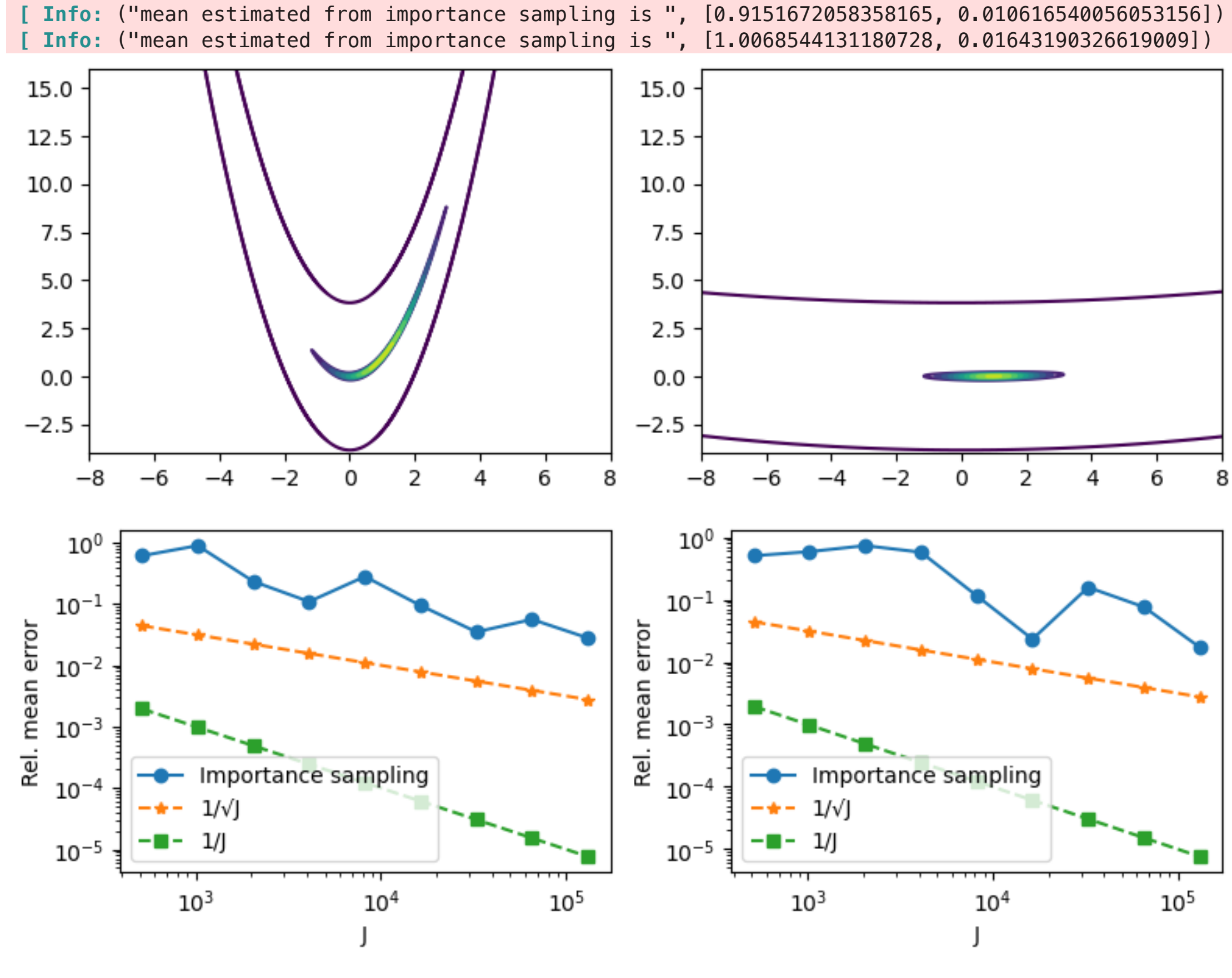
练习（Rosenbrock 函数）

我们要采样的后验分布满足

$$\begin{aligned}\rho_{\mathrm{post}}(\theta) &\propto e^{-\Phi(\theta)}\rho_{\mathrm{prior}}\\ \Phi &= \frac{1}{2}\Big((y - \mathcal{G}(\theta))^T\Sigma_\eta^{-1}(y - \mathcal{G}(\theta))\Big)\\ &= \frac{1}{2}\Big(100(\theta_2 - c_1\theta_1^2)^2 + (1 - \theta_1)^2\Big)\end{aligned}$$

```
using PyPlot
using Random
using Distributions
using LinearAlgebra
Random.seed!(42)

function  Phi_R_Rosenbrock(θ₁, θ₂, c₁, c₂)
    return (100*(θ₂ - c₁*θ₁^2)^2/c₂ + (1.0 - θ₁)^2/c₂  + θ₁^2/100 + θ₂^2/100)/2.0
end

function  Phi_Rosenbrock(θ₁, θ₂, c₁, c₂)
    return (100*(θ₂ - c₁*θ₁^2)^2/c₂ + (1.0 - θ₁)^2/c₂)/2.0
end

Lx , Ux = -8.0,  8.0
Ly , Uy = -4.0,  16
N = 5000
X = zeros(N, N)
Y = zeros(N, N)
ρ = zeros(N, N)
for i = 1:N
    for j = 1:N
        X[i, j], Y[i, j] = Lx + (Ux - Lx) * (i-1)/(N-1), Ly + (Uy - Ly) * (j-1)/(N-1)
    end
end
Δx = Δy = 1/(N-1)

fig, ax = PyPlot.subplots(ncols=2, nrows=1, sharex=false, sharey=false, figsize=(8,3))
fig_error, ax_error = PyPlot.subplots(ncols=2, nrows=1, sharex=false, sharey=false, figsize=(8,3))

for k =1:2
    c₁, c₂=10^(-(2k-2.0)), 1.0
    for i = 1:N
        for j = 1:N
            ρ[i, j] = Phi_R_Rosenbrock(X[i, j], Y[i, j], c₁, c₂)
        end
    end

    ρ .= exp.(-ρ)
    Z = sum(ρ)
    ρ ./= Z
    ρ ./= (Δx * Δy)
    ax[k].contour(X, Y, ρ, 10)


    mean_ref = [sum(X.*ρ)/sum(ρ); sum(Y.*ρ)/sum(ρ)]
    @info "mean = ", mean_ref

    Js = [2^i for i=9:17]
    errors = zeros(length(Js))

    for (J_ind, J) in enumerate(Js)
        θ = 10*rand(Normal(0,1), J, 2)
        Φ = zeros(J)
        for j = 1:J
           Φ[j] =  Phi_Rosenbrock(θ[j,1], θ[j,2], c₁, c₂)
        end
        Φ .-= minimum(Φ)
        w = zeros(J)
        for j = 1:J
           w[j] =  exp(-Φ[j])
        end
        w ./= sum(w)
        mean_IS = (w'*θ)[:]
        @info "mean estimated from importance sampling is ", mean_IS
        errors[J_ind] = norm( mean_IS - mean_ref)/norm(mean_ref)
    end

    ax_error[k].loglog(Js,errors, "-o", label="Importance sampling")
    ax_error[k].loglog(Js,1 ./ sqrt.(Js), "--*", label="1/√J")
    ax_error[k].loglog(Js,1 ./ Js, "--s", label="1/J")
    ax_error[k].set_xlabel("J")
    ax_error[k].set_ylabel("Rel. mean error")
    ax_error[k].legend()
end



fig.tight_layout()
fig.savefig("Rosenbrock_IS.pdf")

fig_error.tight_layout()
fig_error.savefig("Rosenbrock_IS_error.pdf")
```

```
[ Info: ("mean = ", [0.9253911747282657, 1.7546848494609977])
[ Info: ("mean estimated from importance sampling is ", [1.1536540161988076, 2.9427162674561673])
[ Info: ("mean estimated from importance sampling is ", [1.6183534262125314, 3.3842079347340444])
[ Info: ("mean estimated from importance sampling is ", [0.9740422857378545, 1.295450126480252])
[ Info: ("mean estimated from importance sampling is ", [1.013929307615152, 1.9520271707397683])
[ Info: ("mean estimated from importance sampling is ", [0.6260750103647701, 1.2869634709465982])
[ Info: ("mean estimated from importance sampling is ", [1.0611626800811127, 1.8831355926008218])
[ Info: ("mean estimated from importance sampling is ", [0.8613556786227254, 1.7279082871394913])
[ Info: ("mean estimated from importance sampling is ", [0.9211432842716867, 1.6437277983438383])
[ Info: ("mean estimated from importance sampling is ", [0.9580238977313262, 1.7993644136934186])
[ Info: ("mean = ", [0.9900912653489645, 0.019701711185840336])
[ Info: ("mean estimated from importance sampling is ", [1.4754644788871807, -0.09947399502669997])
[ Info: ("mean estimated from importance sampling is ", [0.4448227519244443, 0.2212340463045559])
[ Info: ("mean estimated from importance sampling is ", [0.2690468318133965, -0.06726089411892272])
[ Info: ("mean estimated from importance sampling is ", [0.4215097604208212, -0.011715425476674044])
[ Info: ("mean estimated from importance sampling is ", [1.104592360775503, 0.02254403697972934])
[ Info: ("mean estimated from importance sampling is ", [1.0122388378598166, 0.026074667565751337])
[ Info: ("mean estimated from importance sampling is ", [0.8344618358416795, 0.008600316664862207])
[ Info: ("mean estimated from importance sampling is ", [0.9151672058358165, 0.010616540056053156])
[ Info: ("mean estimated from importance sampling is ", [1.0068544131180728, 0.01643190326619009])
```



3. 无迹变换

对于无迹变换，对 $1 \leq j \leq N_\theta$

$$W_j^m = W_{j+N_\theta}^m \qquad W_j^c = W_{j+N_\theta}^c = \frac{1}{2c_j^2} \qquad \sum_{i=0}^{2N_\theta} W_i^m = 1$$

使用Taylor 展开，我们有

$$\begin{aligned}\mathcal{G}(\theta) =& \mathcal{G}(m) + \nabla\mathcal{G}\delta\theta + \frac{1}{2}\nabla^2\mathcal{G}\delta\theta\otimes\delta\theta + \frac{1}{6}\nabla^3\mathcal{G}\delta\theta\otimes\delta\theta\otimes\delta\theta + O(\delta\theta^4)\\ \mathbb{E}[\mathcal{G}(\theta)] =& \mathcal{G}(m) + \frac{1}{2}\nabla^2\mathcal{G}C + O(\delta\theta^4)\\ \mathrm{Cov}[\mathcal{G}_1(\theta), \mathcal{G}_2(\theta)] =& \mathbb{E}\Big[\Big(\nabla\mathcal{G}_1\delta\theta + \frac{1}{2}\nabla^2\mathcal{G}_1\delta\theta\otimes\delta\theta + \frac{1}{6}\nabla^3\mathcal{G}_1\delta\theta\otimes\delta\theta\otimes\delta\theta - \frac{1}{2}\nabla^2\mathcal{G}_1 C + O(\delta\theta^4)\Big)\\ &\Big(\nabla\mathcal{G}_2\delta\theta + \frac{1}{2}\nabla^2\mathcal{G}_2\delta\theta\otimes\delta\theta + \frac{1}{6}\nabla^3\mathcal{G}_2\delta\theta\otimes\delta\theta\otimes\delta\theta - \frac{1}{2}\nabla^2\mathcal{G}_2 C + O(\delta\theta^4)\Big)^T\\ =& \nabla\mathcal{G}_1 C\nabla\mathcal{G}_2^T + O(\delta\theta^4)\end{aligned}$$

其中 $\nabla^k\mathcal{G}$ 是在 m 的取值，$\delta\theta = \theta - m$, 我们认为 $C = O(\delta\theta^2)$。对于期望的近似，我们有

$$\begin{aligned}\widehat{\mathbb{E}[\mathcal{G}]} =& W_0^m\mathcal{G}(m) + \sum_{j=1}^{N_\theta}\big(W_j^m\mathcal{G}(m + c_j[\sqrt{C}]_j) + W_{j+N_\theta}^m\mathcal{G}(m - c_j[\sqrt{C}]_j)\big)\\ =& W_0^m\mathcal{G}(m) + \sum_{j=1}^{N_\theta}\Big(W_j^m\big(\mathcal{G}(m) + c_j\nabla\mathcal{G}[\sqrt{C}]_j + \frac{c_j^2}{2}\nabla^2\mathcal{G}[\sqrt{C}]_j\otimes[\sqrt{C}]_j + \frac{c_j^3}{6}\nabla^3\mathcal{G}[\sqrt{C}]_j\otimes[\sqrt{C}]_j\otimes[\sqrt{C}]_j + O(\delta\theta^4)\big)\\ &+ W_{j+N_\theta}^m\big(\mathcal{G}(m) - c_j\nabla\mathcal{G}[\sqrt{C}]_j + \frac{c_j^2}{2}\nabla^2\mathcal{G}[\sqrt{C}]_j\otimes[\sqrt{C}]_j - \frac{c_j^3}{6}\nabla^3\mathcal{G}[\sqrt{C}]_j\otimes[\sqrt{C}]_j\otimes[\sqrt{C}]_j + O(\delta\theta^4)\big)\Big)\\ =& \mathcal{G}(m) + \sum_{j=1}^{N_\theta}\frac{c_j^2}{2}(W_j^m + W_{j+N_\theta}^m)\nabla^2\mathcal{G}[\sqrt{C}]_j\otimes[\sqrt{C}]_j + O(\delta\theta^4)) \qquad \text{使用} W_j^m = W_{j+N_\theta}^m\end{aligned}$$

定义 $P = \sum_{j=1}^{N_\theta}\frac{c_j^2}{2}(W_j^m + W_{j+N_\theta}^m)[\sqrt{C}]_j\otimes[\sqrt{C}]_j$， 对于方差的近似，我们有

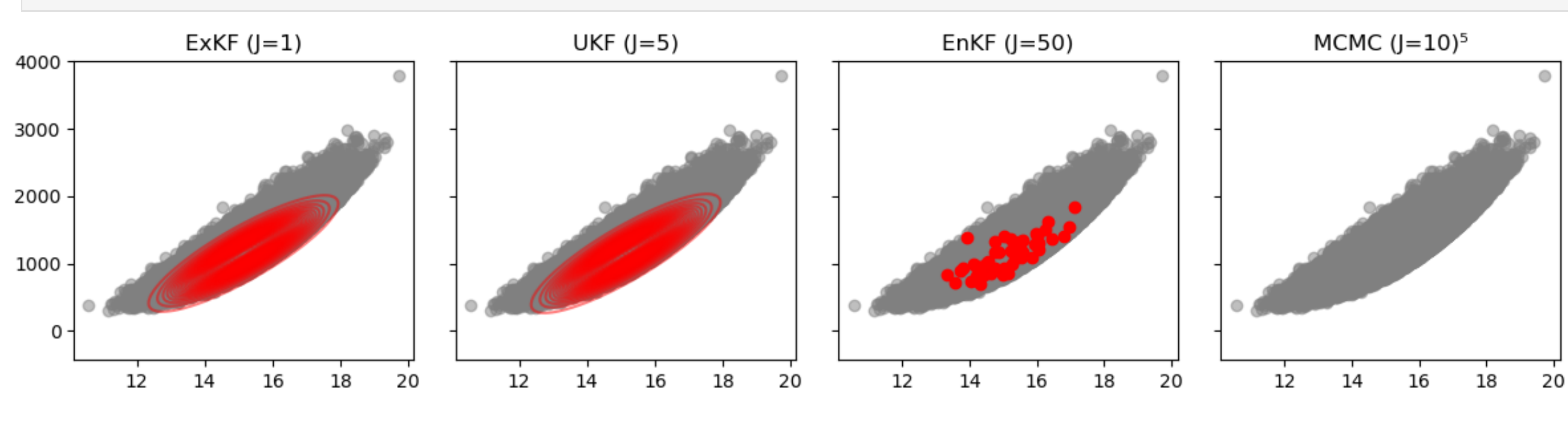
$$\begin{aligned}&W_0^c(\mathcal{G}_1(m) - \widehat{\mathbb{E}[\mathcal{G}_1]})(\mathcal{G}_2(m) - \widehat{\mathbb{E}[\mathcal{G}_2]})^T + \sum_{j=1}^{N_\theta} W_j^c(\mathcal{G}_1(\theta^j) - \widehat{\mathbb{E}[\mathcal{G}_1]})(\mathcal{G}_2(\theta^j) - \widehat{\mathbb{E}[\mathcal{G}_2]})^T\\ &+ W_{j+N_\theta}^c(\mathcal{G}_1(\theta^j) - \widehat{\mathbb{E}[\mathcal{G}_1]})(\mathcal{G}_2(\theta^j) - \widehat{\mathbb{E}[\mathcal{G}_2]})^T\\ =& W_0^c\nabla^2\mathcal{G}_1 PP^T\nabla^2\mathcal{G}_2^T +\\ &\sum_{j=1}^{N_\theta}\Big(W_j^c\big(c_j\nabla\mathcal{G}_1[\sqrt{C}]_j + \nabla^2\mathcal{G}_1(\frac{c_j^2}{2}[\sqrt{C}]_j\otimes[\sqrt{C}]_j - P)\big)\big(c_j\nabla\mathcal{G}_2[\sqrt{C}]_j + \nabla^2\mathcal{G}_2(\frac{c_j^2}{2}[\sqrt{C}]_j\otimes[\sqrt{C}]_j - P)\big)^T\\ &+ W_{j+N_\theta}^c\big(-c_j\nabla\mathcal{G}_1[\sqrt{C}]_j + \nabla^2\mathcal{G}_1(\frac{c_j^2}{2}[\sqrt{C}]_j\otimes[\sqrt{C}]_j - P)\big)\big(-c_j\nabla\mathcal{G}_2[\sqrt{C}]_j + \nabla^2\mathcal{G}_2(\frac{c_j^2}{2}[\sqrt{C}]_j\otimes[\sqrt{C}]_j - P)\big)^T\Big)\\ &+ O(\delta\theta^4)\\ =& \sum_{j=1}^{N_\theta} 2c_j^2 W_j^c\nabla\mathcal{G}_1[\sqrt{C}]_j\otimes[\sqrt{C}]_j\nabla\mathcal{G}_2^T + O(\delta\theta^4) \qquad \text{使用} W_j^c = W_{j+N_\theta}^c\\ =& \nabla\mathcal{G}_1 C\nabla\mathcal{G}_2^T + O(\delta\theta^4) \qquad \text{使用} W_j^c = W_{j+N_\theta}^c = \frac{1}{2c_j^2}\end{aligned}$$

4. 卡尔曼变换

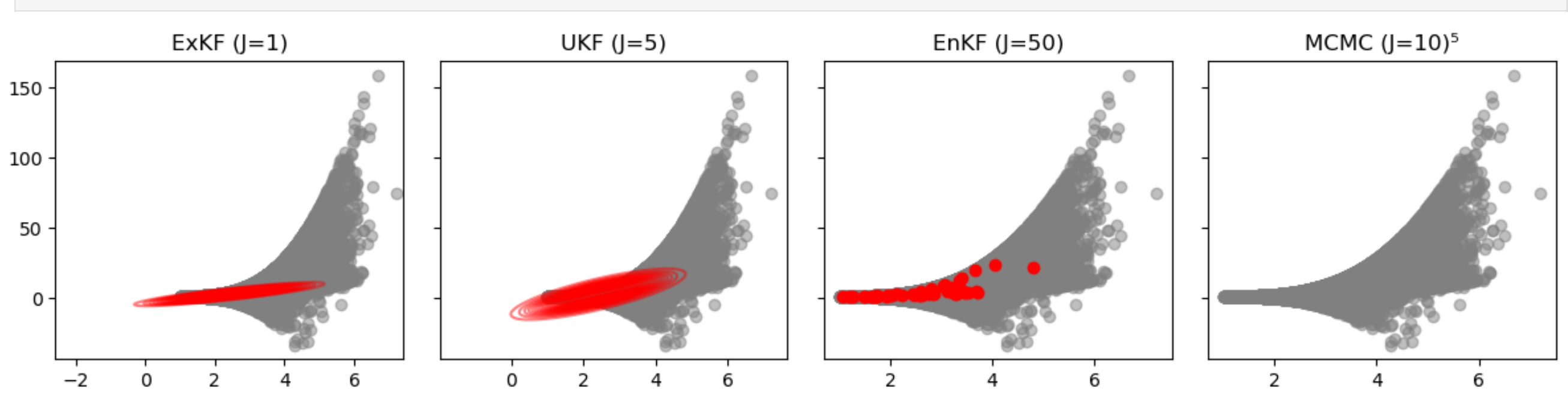
```
using Random, Distributions, LinearAlgebra, PyPlot


function Gaussian_2d(θ_mean::Array{FT,1}, θθ_cov, Nx::IT, Ny::IT, x_min=nothing, x_max=nothing, y_min=nothing, y_max=
    # 2d Gaussian plot


    if x_min === nothing
        x_min = θ_mean[1] - 5*sqrt(θθ_cov[1,1])
    end
    if x_max === nothing
        x_max = θ_mean[1] + 5*sqrt(θθ_cov[1,1])
    end
    if y_min === nothing
        y_min = θ_mean[2] - 5*sqrt(θθ_cov[2,2])
    end
    if y_max === nothing
        y_max = θ_mean[2] + 5*sqrt(θθ_cov[2,2])
    end


    xx = Array(LinRange(x_min, x_max, Nx))
    yy = Array(LinRange(y_min, y_max, Ny))

    X,Y = repeat(xx, 1, Ny), repeat(yy, 1, Nx)'
    Z = zeros(FT, Nx, Ny)

    det_θθ_cov = det(θθ_cov)

    for ix = 1:Nx
        for iy = 1:Ny
            Δxy = [xx[ix] - θ_mean[1]; yy[iy] - θ_mean[2]]
            Z[ix, iy] = exp(-0.5*(Δxy'/θθ_cov*Δxy)) / (2 * pi * sqrt(det_θθ_cov))
        end
    end

    return X, Y, Z

end



function ExKF(G::Function, m::Array{Float64,1}, C::Array{Float64,2})
    g, dg = G(m)
    mg, Cg = g, dg*C*dg'

    return mg, Cg
end


function UKF(G::Function, m::Array{Float64,1}, C::Array{Float64,2})
    N_θ = length(m)
    N_ens = 2N_θ + 1

    # weights
    a = min(sqrt(4/(N_θ)), 1.0)
    λ = a^2*(N_θ) - N_θ
    sigma_weight = sqrt(N_θ + λ)
    cov_weight = 1/(2(N_θ + λ))

    # generate sigma points
    chol_C = cholesky(Hermitian(C)).L
    θ = zeros(Float64, N_θ, N_ens)
    θ[:, 1] = m
    for i = 1: N_θ
        θ[:, i+1] = m + sigma_weight*chol_C[:,i]
        θ[:, i+1+N_θ] = m - sigma_weight*chol_C[:,i]
    end

    g = zeros(Float64, N_θ, N_ens)
    for i = 1: 2N_θ+1
        g[:, i], _ = G(θ[:, i])
    end

    # compute mean
    mg = g[:, 1]

    Cg = zeros(Float64, N_θ, N_θ)
    for i = 1: N_ens
        Cg .+= cov_weight*(g[:, i] - mg)*(g[:, i] - mg)'
    end

    return mg, Cg
end


function EKF(G::Function, m::Array{Float64,1}, C::Array{Float64,2}, J::Int64)
    N_θ = length(m)

    θ = rand(MvNormal(m, C), J)

    g = zeros(Float64, N_θ, J)
    for i = 1: J
        g[:, i], _ = G(θ[:, i])
    end


    mg = sum(g, dims=2)/J

    Cg = zeros(Float64, N_θ, N_θ)
    for i = 1: J
        Cg .+= (g[:, i] - mg)*(g[:, i] - mg)'
    end
    Cg ./= (J - 1)

    return mg, Cg, g

end


function G(θ)
    g = [1 + sqrt(θ[1]^2 + θ[2]^2); exp(θ[1]/2) + θ[2]^3]
    dg = [θ[1]/(sqrt(θ[1]^2 + θ[2]^2)) θ[2]/(sqrt(θ[1]^2 + θ[2]^2));
          exp(θ[1]/2)/2                3θ[2]^2]
    return g, dg
end
```

```
EKF (generic function with 1 method)
```

```
m, C = [10.0; 10.0], [1.0^2 0.0; 0.0 1.0^2]

mckf = EKF(G, m, C, 100000)
exkf = ExKF(G, m, C)
ukf = UKF(G, m, C)
ekf = EKF(G, m, C, 50)





Nx = Ny = 100

fig, ax = PyPlot.subplots(ncols=4, sharex=false, sharey=true, figsize=(12,3))
for i = 1:4
    ax[i].scatter(mckf[3][1, :], mckf[3][2, :], color = "grey", alpha=0.5, label = "MCMC")
end

X,Y,Z = Gaussian_2d(exkf[1], exkf[2], Nx, Ny)
ax[1].contour(X, Y, Z, 50, colors = "red", alpha=0.5, label = "ExKF (J=1)")
ax[1].set_title("ExKF (J=1)")

X,Y,Z = Gaussian_2d(ukf[1], ukf[2], Nx, Ny)
ax[2].contour(X, Y, Z, 50, colors = "red", alpha=0.5, label = "UKF (J=5)")
ax[2].set_title("UKF (J=5)")

ax[3].scatter(ekf[3][1, :], ekf[3][2, :], color = "red", label = "EKF")
ax[3].set_title("EnKF (J=50)")

ax[4].set_title("MCMC (J=10)⁵")

fig.tight_layout()
fig.savefig("KM-1.png")
```


ExKF (J=1) UKF (J=5) EnKF (J=50) MCMC (J=10)⁵

```
m, C = [1.0; 1.0], [1.0^2 0.0; 0.0 1.0^2]

mckf = EKF(G, m, C, 100000)
exkf = ExKF(G, m, C)
ukf = UKF(G, m, C)
ekf = EKF(G, m, C, 50)


Nx = Ny = 100

fig, ax = PyPlot.subplots(ncols=4, sharex=false, sharey=true, figsize=(12,3))
for i = 1:4
    ax[i].scatter(mckf[3][1, :], mckf[3][2, :], color = "grey", alpha=0.5, label = "MCMC")
end

X,Y,Z = Gaussian_2d(exkf[1], exkf[2], Nx, Ny)
ax[1].contour(X, Y, Z, 50, colors = "red", alpha=0.5, label = "ExKF (J=1)")
ax[1].set_title("ExKF (J=1)")

X,Y,Z = Gaussian_2d(ukf[1], ukf[2], Nx, Ny)
ax[2].contour(X, Y, Z, 50, colors = "red", alpha=0.5, label = "UKF (J=5)")
ax[2].set_title("UKF (J=5)")

ax[3].scatter(ekf[3][1, :], ekf[3][2, :], color = "red", label = "EKF")
ax[3].set_title("EnKF (J=50)")

ax[4].set_title("MCMC (J=10)⁵")

fig.tight_layout()
fig.savefig("KM-2.png")
```


ExKF (J=1) UKF (J=5) EnKF (J=50) MCMC (J=10)⁵

5. 实值非体积保持模型

对于仿射耦合层

$$\begin{aligned}y_{1:d} &= x_{1:d}\\ y_{d+1:N_\theta} &= x_{d+1:N_\theta}\odot\exp(s(x_{1:d})) + t(x_{1:d})\end{aligned}$$

可逆

$$\begin{aligned}x_{1:d} &= y_{1:d}\\ x_{d+1:N_\theta} &= \exp(-s(x_{1:d}))\odot\big(y_{d+1:N_\theta} - t(x_{1:d})\big)\end{aligned}$$

微分同胚

$$\frac{\partial y}{\partial x} = \begin{bmatrix} I & 0\\ \frac{\partial t}{\partial x_{1:d}} + \frac{\partial s}{\partial x_{1:d}}\odot\big(x_{d+1:N_\theta}\odot\exp(s(x_{1:d}))\big) & \exp(s(x_{1:d}))\end{bmatrix}$$